

**SMC** 2019  
MÁLAGA, Spain

28 - 31 May

**16th Sound and Music Computing Conference**

**Proceedings**





# **SMC 2019**

**Proceedings**  
**of the**  
**16th Sound & Music**  
**Computing Conference**



**May 28 - 31, 2019**  
**Málaga, Spain**



SMC 2019 is organized by

**ATIC Research Group**  
**Universidad de Málaga**



Website: <http://smc2019.uma.es>

## **Proceedings of the 16th Sound & Music Computing Conference** **SMC 2019**

Cover design by Alberto Peinado, Isabel Barbancho & Lorenzo J. Tardón  
SMC 2019 logo by Alberto Peinado

Edited by:

Isabel Barbancho (ATIC Research Group. Universidad de Málaga)  
Lorenzo J. Tardón (ATIC Research Group. Universidad de Málaga)  
Alberto Peinado (Universidad de Málaga)  
Ana M. Barbancho (ATIC Research Group. Universidad de Málaga)

ISBN 978-84-09-08518-7  
ISSN 2518-3672

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2019 SMC 2019

## ORGANIZATION



GRUPO DE APLICACIÓN DE LAS  
TECNOLOGÍAS DE LA INFORMACIÓN  
Y COMUNICACIONES



UNIVERSIDAD  
DE MÁLAGA

| **uma.es**



E.T.S. DE INGENIERÍA DE  
**TELECOMUNICACIÓN**  
UNIVERSIDAD DE MÁLAGA





## SPONSORS



UNIVERSIDAD  
DE MÁLAGA



Vicerrectorado de Investigación y Transferencia



*applied sciences*

an Open Access Journal by MDPI



**audio-technica**



**Fundación  
Unicaja**

Other collaborators



---

# Contents

<b>Contents</b>	<b>v</b>
<b>Preface</b>	<b>1</b>
<b>SMC 2019 Conference Committee</b>	<b>3</b>
<b>SMC 2019 Contributions</b>	<b>5</b>
P1. Poster Session 1 . . . . .	7
P1.1 DAW-Integrated Beat Tracking for Music Production <i>Brett Dalton, David Johnson and George Tzanetakis</i> . . . . .	7
P1.2 Interaction-Based Analysis of Freely Improvised Music <i>Stefano Kalonaris</i> . . . . .	12
P1.3 Mechanical Entanglement: A Collaborative Haptic-Music Performance <i>Alexandros Kontogeorgakopoulos, George Sioros and Odysseas Klissouras</i> . . . . .	20
P1.4 State Dependency - Audiovisual Interaction through Brain States <i>Patrick Neff, Jan Schacher and Daniel Bisig</i> . . . . .	26
P1.5 Perceptual Evaluation of Modal Synthesis for Impact-Based Sounds <i>Adrián Barahona and Sandra Pauletto</i> . . . . .	34
P1.6 VIBRA - Technical and Artistic Issues in an Interactive Dance Project <i>Andreas Bergsland, Sigurd Saue and Pekka Stokke</i> . . . . .	39
P1.7 Musical Tempo and Key Estimation using Convolutional Neural Networks with Directional Filters <i>Hendrik Schreiber and Meinard Müller</i> . . . . .	47
P1.8 The Viking HRTF Dataset <i>Simone Spagnol, Kristján Bjarki Purkhús, Runar Unnthórsson and Sverrir Karl Björnsson</i> . . . . .	55
P1.9 Performing with Sound Sample-Controlled Gloves and Light-Controlled Arms <i>Justin Pecquet, Fotis Moschos, David Fierro and Frank Pecquet</i> . . . . .	61
P1.10 Melody Identification in Standard MIDI Files <i>Zheng Jiang and Roger Dannenberg</i> . . . . .	65
P1.11 Automatic Chord-Scale Recognition using Harmonic Pitch Class Profiles <i>Emir Demirel, Baris Bozkurt and Xavier Serra</i> . . . . .	72
D1. Demo Session 1 . . . . .	80
D1.1 Interacting with Digital Resonators by Acoustic Excitation <i>Max Neupert and Clemens Wegener</i> . . . . .	80
D1.2 Melody Slot Machine <i>Masatoshi Hamanaka</i> . . . . .	82



D1.3 OM-AI: A Toolkit to Support AI-Based Computer-Assisted Composition Workflows in Open-Music <i>Anders Vinjar and Jean Bresson</i>	84
D1.4 URALi: a Proposal of Approach to Real-Time Audio Synthesis in Unity <i>Enrico Dorigatti</i>	86
D1.5 A Sequencer with Decoupled Track Timing <i>Silvan David Peter and Gerhard Widmer</i>	88
D1.6 Musicypher: Music for Message Encryption <i>Víctor Jaime and Alberto Peinado</i>	89
D1.7 A Platform for Processing Sheet Music and Developing Multimedia Application <i>Fu-Hai Frank Wu</i>	91
D1.8 Capturing the Reaction Time to Distinguish between Voice and Music <i>Alejandro Villena-Rodríguez, Lorenzo J. Tardón, Isabel Barbancho, Ana M. Barbancho, Irene Gómez-Plazas and María-José Varela-Salinas</i>	93
D1.9 Physical Models and Real-Time Control with the Sensel Morph <i>Silvin Willemsen, Stefan Bilbao, Nikolaj Andersson and Stefania Serafin</i>	95
S1. Oral Session 1. Sonic Interactions	97
S1.1 Towards a High-Performance Platform for Sonic Interaction Interfaces <i>Stefano Fasciani and Manohar Vohra</i>	97
S1.2 Digital Manufacturing For Musical Applications: A Survey of Current Status and Future Outlook <i>Doga Cavdir</i>	105
S1.3 Real Time Audio Digital Signal Processing with Faust and the Teensy <i>Romain Michon, Yann Orlarey, Stéphane Letz and Dominique Fober</i>	112
S1.4 Sound Design through Large Audience Interaction <i>Kjetil Falkenberg Hansen, Martin Ljungdahl-Eriksson and Ricardo Atienza</i>	119
S1.5 Evaluating a Continuous Sonic Interaction: Comparing a Performable Acoustic and Digital Everyday Sound <i>Fiona Keenan and Sandra Pauletto</i>	127
S2. Oral Session 2. Nordic SMC	135
S2.1 Adaptive Loudness Compensation in Music Listening <i>Leonardo Fierro, Jussi Rämö and Vesa Välimäki</i>	135
S2.2 Toward Automatic Tuning of the Piano <i>Joonas Tuovinen, Jamin Hu and Vesa Välimäki</i>	143
S2.3 Real-time Control of Large-scale Modular Physical Models using the Sensel Morph <i>Silvin Willemsen, Nikolaj Andersson, Stefania Serafin and Stefan Bilbao</i>	151
S2.4 An Interactive Music Synthesizer for Gait Training in Neurorehabilitation <i>Prithvi Kantan and Sofia Dahl</i>	159
S2.5 From Vocal Sketching to Sound Models by Means of a Sound-Based Musical Transcription System <i>Claudio Panariello, Mattias Sköld, Emma Frid and Roberto Bresin</i>	167
S2.6 Tempo and Metrical Analysis by Tracking Multiple Metrical Levels Using Autocorrelation <i>Olivier Lartillot and Didier Grandjean</i>	174
S3. Oral Session 3. Augmented and Virtual Realities	182
S3.1 Comparison and Implementation of Data Transmission Techniques through Analog Audio Signals in the Context of Augmented Mobile Instruments <i>Romain Michon, Yann Orlarey, Stéphane Letz and Dominique Fober</i>	182
S3.2 Mass-Interaction Physical Models for Sound and Multi-Sensory Creation: Starting Anew <i>Jerome Villeneuve and James Leonard</i>	187

S3.3 Exploring the Effects of Diegetic and Non-diegetic Audiovisual Cues on Decision-making in Virtual Reality <i>Anil Çamcı</i>	195
S3.4 OSC-XR: A Toolkit for Extended Reality Immersive Music Interfaces <i>David Johnson, Daniela Damian and George Tzanetakis</i>	202
S3.5 No Strings Attached: Force and Vibrotactile Feedback in a Guitar Simulation <i>Andrea Passalenti, Razvan Paisa, Niels C. Nilsson, Nikolaj S. Andersson, Federico Fontana, Rolf Nordahl and Stefania Serafin</i>	210
P2. Poster Session 2	217
P2.1 RaveForce: A Deep Reinforcement Learning Environment for Music Generation <i>Qichao Lan, Jim Tørresen and Alexander Refsum Jensenius</i>	217
P2.2 Music Temperaments Evaluation Based on Triads <i>Tong Meihui and Satoshi Tojo</i>	223
P2.3 Composing Space in the Space: An augmented and Virtual Reality Sound Spatialization System <i>Giovanni Santini</i>	229
P2.4 Graph Based Physical Models for Sound Synthesis <i>Pelle Juul Christensen and Stefania Serafin</i>	234
P2.5 ADEPT: Exploring the Design, Pedagogy and Analysis of a Mixed Reality Application for Piano Training <i>Lynda Joy Gerry, Sofia Dahl and Stefania Serafin</i>	241
P2.6 A Model Comparison for Chord Prediction on the Annotated Beethoven Corpus <i>Kristoffer Landsnes, Liana Mehrabyan, Victor Wiklund, Robert Lieck, Fabian C. Moss and Martin Rohrmeier</i>	250
P2.7 Sonic Characteristics of Robots in Films <i>Adrian B. Latupeirissa, Emma Frid and Roberto Bresin</i>	255
P2.8 Virtual Reality Music Intervention to Reduce Social Anxiety in Adolescents Diagnosed with Autism Spectrum Disorder <i>Ali Adjorlu, Nathaly Belen Betancourt Barriga and Stefania Serafin</i>	261
P2.9 Teach Me Drums: Learning Rhythms through the Embodiment of a Drumming Teacher in Virtual Reality <i>Mie Moth-Poulsen, Tomasz Bednarz, Volker Kuchelmeister and Stefania Serafin</i>	269
P2.10 Real-time Mapping of Periodic Dance Movements to Control Tempo in Electronic Dance Music <i>Lilian Jap and Andre Holzapfel</i>	274
P2.11 Increasing Access to Music in SEN Settings <i>Tom Davis, Daniel Pierson and Ann Bevan</i>	281
D2. Demo Session 2	287
D2.1 Interacting with Musebots (that don't really listen) <i>Arne Eigenfeldt</i>	287
D2.2 Extending Jamsketch: An Improvisation Support System <i>Akane Yasuhara, Junko Fujii and Tetsuro Kitahara</i>	289
D2.3 Visualizing Music Genres using a Topic Model <i>Swaroop Panda, Vinay P. Nambodiri and Shatarupa Thakurta Roy</i>	291
D2.4 CompoVOX: Real-Time Sonification of Voice <i>Daniel Hernán Molina Villota, Isabel Barbancho and Antonio Jurado-Navas</i>	293
D2.5 Facial Activity Detection to Monitor Attention and Fatigue <i>Oscar Cobos, Jorge Munilla, Ana M. Barbancho, Isabel Barbancho and Lorenzo J. Tardón</i>	295
D2.6 The Chordinator: An Interactive Music Learning Device <i>Eamon McCoy, John Greene, Jared Henson, James Pinder, Jonathon Brown and Claire Arthur</i>	297



D2.7 Automatic Chord Recognition in Music Education Applications <i>Sascha Grollmisch and Estefania Cano</i>	299
D2.8 Sonic Sweetener Mug <i>Signe Lund Mathiesen, Derek Victor Byrne and Qian Janice Wang</i>	300
S4. Oral Session 4. SMC Tools and Methodologies	302
S4.1 A Framework for the Development and Evaluation of Graphical Interpolation for Synthesizer Parameter Mappings <i>Darrell Gibson and Richard Polfreman</i>	302
S4.2 Composing with Sounds: Designing an Object Oriented DAW for the Teaching of Sound- Based Composition <i>Stephen Pearse, Leigh Landy, Duncan Chapman, David Holland and Mihai Eni</i>	310
S4.3 Insights in Habits and Attitudes Regarding Programming Sound Synthesizers: A Quantitative Study <i>Gordan Kreković</i>	316
S5. Oral Session 5. Sound Synthesis & Analysis	324
S5.1 Experimental Verification of Dispersive Wave Propagation on Guitar Strings <i>Dmitri Kartofelev, Joann Gustav Arro and Vesa Välimäki</i>	324
S5.2 Real-Time Modeling of Audio Distortion Circuits with Deep Learning <i>Eero-Pekka Damskögg, Lauri Juvela and Vesa Välimäki</i>	332
S5.3 MI-GEN~: An Efficient and Accessible Mass-Interaction Sound Synthesis Toolbox <i>James Leonard and Jerome Villeneuve</i>	340
S5.4 Combining Texture-Derived Vibrotactile Feedback, Concatenative Synthesis and Photogram- metry for Virtual Reality Rendering <i>Eduardo Magalhães, Emil Rosenlund Høeg, Gilberto Bernardes, Jon Ram Bruun-Pedersen, Stefania Serafin and Rolf Nordhal</i>	348
S5.5 Percussion Synthesis using Loopback Frequency Modulation Oscillators <i>Jennifer Hsu and Tamara Smyth</i>	356
S5.6 Deep Linear Autoregressive Model for Interpretable Prediction of Expressive Tempo <i>Akira Maezawa</i>	364
S5.7 Metrics for the Automatic Assessment of Music Harmony Awareness in Children <i>Federico Avanzini, Adriano Baratè, Luca Andrea Ludovico and Marcella Mandanici</i>	372
S6. Oral Session 6. Music Information Processing	380
S6.1 Learning to Generate Music with BachProp <i>Florian Colombo, Johanni Brea and Wulfram Gerstner</i>	380
S6.2 Offline Score Alignment for Realistic Music Practice <i>Yucong Jiang, Fiona Ryan, David Cartledge and Christopher Raphael</i>	387
S6.3 Piano Score-Following by Tracking Note Evolution <i>Yucong Jiang and Christopher Raphael</i>	394
S6.4 Adaptive Score-Following System by Integrating Gaze Information <i>Kaede Noto, Yoshinari Takegawa and Keiji Hirata</i>	401
S6.5 Alternative Measures: A Musicologist Workbench for Popular Music <i>Beach Clark and Claire Arthur</i>	407
P3. Poster Session 3	415
P3.1 Autoencoders for Music Sound Modeling: a Comparison of Linear, Shallow, Deep, Recurrent and Variational Models <i>Fanny Roche, Thomas Hueber, Samuel Limier and Laurent Girin</i>	415

P3.2 Polytopic Reconfiguration: a Graph-Based Scheme for the Multiscale Transformation of Music Segments and its Perceptual Assessment <i>Valentin Gillot and Frédéric Bimbot</i>	423
P3.3 Non-linear Contact Sound Synthesis for Real-Time Audiovisual Applications using Modal Textures <i>Martin Maunsbach and Stefania Serafin</i>	431
P3.4 Analysis of Vocal Ornamentation in Iranian Classical Music <i>Sepideh Shafiei</i>	437
P3.5 VUSAA: An Augmented Reality Mobile App for Urban Soundwalks <i>Josué Moreno and Vesa Norilo</i>	442
P3.6 A Framework for Multi-f0 Modeling in SATB Choirs <i>Helena Cuesta, Emilia Gómez and Pritish Chandna</i>	447
P3.7 Representations of Self-Coupled Modal Oscillators with Time-Varying Frequency <i>Tamara Smyth and Jennifer Hsu</i>	454
P3.8 SonaGraph. A Cartoonified Spectral Model for Music Composition <i>Andrea Valle</i>	462
P3.9 Sound in Multiples: Synchrony and Interaction Design using Coupled-Oscillator Networks <i>Nolan Lem</i>	470
P3.10 ‘Jazz Mapping’ an Analytical and Computational Approach to Jazz Improvisation <i>Dimitrios Vassilakis, Anastasia Georgaki and Christina Anagnostopoulou</i>	476
P3.11 Visual Pitch Estimation <i>A. Sophia Koepke, Olivia Wiles and Andrew Zisserman</i>	483
D3. Demo Session 3	489
D3.1 MiningSuite: A Comprehensive Matlab Framework for Signal, Audio and Music Analysis, Articulating Audio and Symbolic Approaches <i>Olivier Lartillot</i>	489
D3.2 Drawing Geometric Figures with Braille Description through a Speech Recognition System <i>África Chamorro, Ana M. Barbancho, Isabel Barbancho and Lorenzo J. Tardón</i>	490
D3.3 Interactive Music Training System <i>Daniel Moreno, Isabel Barbancho, Ana M. Barbancho and Lorenzo J. Tardón</i>	492
D3.4 Copying Clave - A Turing Test <i>Simon Blackmore</i>	494
D3.5 Resonance Improviser: A System for Transmitting the Embodied Sensations of Vocalization between two People during Improvisation <i>Tejaswinee Kelkar and Lynda Joy Gerry</i>	495
D3.6 Finding New Practice Material through Chord-Based Exploration of a Large Music Catalogue <i>Johan Pauwels and Mark B. Sandler</i>	497
D3.7 Internal Complexity for Exploratory Interaction <i>Mads Hoby</i>	499
D3.8 Adaptive Body Movement Sonification in Music and Therapy <i>Christian Baumann, Johanna Friederike Baarlink and Jan-Torsten Milde</i>	501
S7. Oral Session 7. Multimodality and (e)motions	505
S7.1 VocalistMirror: A Singer Support Interface for Avoiding Undesirable Facial Expressions <i>Kin Wah Edward Lin, Tomoyasu Nakano and Masataka Goto</i>	505
S7.2 Audiovisual Perception of Arousal, Valence and Effort in Contemporary Cello Performance <i>Hanna Järveläinen</i>	511
S7.3 Dancing Dots - Investigating the Link between Dancer and Musician in Swedish Folk Dance <i>Olof Misgeld, Andre Holzapfel and Sven Ahlbäck</i>	519

S8. Oral Session 8. Machine Learning . . . . .	525
S8.1 Conditioning a Recurrent Neural Network to Synthesize Musical Instrument Transients <i>Lonce Wyse and Muhammad Huzaifah</i> . . . . .	525
S8.2 Predicting Perceived Dissonance of Piano Chords using a Chord-Class Invariant CNN and Deep Layered Learning <i>Juliette Dubois, Anders Elowsson and Anders Friberg</i> . . . . .	530
S8.3 Belief Propagation Algorithm for Automatic Chord Estimation <i>Vincent P. Martin, Sylvain Reynal, Dogac Basaran and Hélène-Camille Crayencour</i> . . . . .	537
S8.4 HMM-Based Glissando Detection for Recordings of Chinese Bamboo Flute <i>Changhong Wang, Emmanouil Benetos, Xiaojie Meng and Elaine Chew</i> . . . . .	545
S8.5 Towards CNN-based Acoustic Modeling of Seventh Chords for Automatic Chord Recognition <i>Christon-Ragavan Nadar, Jakob Abeßer and Sascha Grollmisch</i> . . . . .	551
S8.6 From Jigs and Reels to Schottisar och Polskor: Generating Scandinavian-like Folk Music with Deep Recurrent Networks <i>Simon Mossmyr, Eric Hallström, Bob L. Sturm, Victor Hansjons Vegeborn and Jonas Wedin</i> . . . . .	558
S8.7 Modeling and Learning Rhythm Structure <i>Francesco Foscarin, Florent Jacquemard and Philippe Rigaux</i> . . . . .	566
<b>Acknowledgments</b>	<b>573</b>
<b>Reviewers</b>	<b>575</b>
<b>Author Index</b>	<b>577</b>

---

# Preface

The Sound and Music Computing Conference reaches its 16th edition!

This 16th Sound & Music Computing Conference (SMC 2019) takes place in Málaga, Spain, May 28-31, 2019 and it is organized by the Application of Information and Communication Technologies Research group (ATIC) of Universidad de Málaga (UMA).

SMC 2019 Topics of Interest include a wide selection of topics related to acoustics, psychoacoustics, music, technology for music, audio analysis, musicology, sonification, music games, machine learning, serious games, immersive audio, sound synthesis, etc.

Each year a specific topic of interest is highlighted. The theme of this year is **Music and Interaction**. This topic is as broad as the different ways in which a certain person (musician or not, engineer or not, artist or not...) may want to interact with music, depending on their personal interests and their specific relationship with music.

The interaction can be focused on music composition and creation, music performance, data mining, the influence of music on human beings, music learning, musical instruments, new sound development, reading music, etc.

SMC 2019 conference fosters the presentation of new methods for any kind of musical conversation or conversation through music, human-computer interaction through the lens of sound, interaction models from objects to bio and AI systems, new interfaces for playing music, interactive content discovery and recommendation, optical music recognition, music education, music games, sonification and any other music information retrieval related technology.

SMC 2019 is an interdisciplinary forum to share research, music, thoughts, needs and discoveries between musicians in a broad sense, computer science experts, music information retrieval researchers, etc. This interdisciplinary atmosphere will be the perfect place to come up with new ideas, applications and challenges to keep on working in this fantastic research topic that brings together art, technology and human perception.

SMC 2019 welcomed different types of contributions:

- **Papers** examining all the core topics of the Sound and Music Computing field; these contributions, that have been fully peer-reviewed, will be presented as oral presentation or poster.
- **Musical contributions** that make use of the possibilities technology offers nowadays to create music in a broad sense. Selected contributions will be performed at the scheduled music sessions.
- **Demos**, that are a novelty this year. Demo contributions, which have been very well accepted by the community, are intended to the presentation of preliminary results, ideas, applications or system prototypes that are not yet fully formed nor systematically evaluated, but of interest to the SMC community.

SMC 2019 received 166 submission: 97 papers, 43 musical contributions and 26 demos. Out of them, SMC 2019 features 41 oral presentations, 33 posters, 24 musical pieces and 25 demos.

SMC 2019 had the help of 117 scientific reviewers and 37 music reviewers to examine all the submissions in order to compile the final Scientific and Music Program. The Program Chairs and the Music Chairs, together with the General Chairs, have made a great job making the final decisions and organizing the presentation of the different contributions in the Oral, Poster, Music and Demo Sessions.

A meet report with the conference abstracts and a cooperated special issue focused on ‘Sound and Music Computing – Music and Interaction’ that will include extended versions of selected contributions to the 2019 Sound and Music Computing Conference will be published by Applied Sciences, an Open Access Journal by MDPI.

In this book, you can find the Proceedings of SMC 2019 with all the scientific contributions presented during the conference.

**SMC 2019 General Chairs***Isabel Barbancho**Lorenzo J. Tardón*

Málaga

May 2019

**SMC 2019 Program Chairs***Stefania Serafin**Federico Avanzini*

---

# SMC 2019 Conference Committee

## **General Chairs**

Isabel Barbancho (Universidad de Málaga, Spain)  
Lorenzo J. Tardón (Universidad de Málaga, Spain)

## **Program Chairs**

Stefania Serafin (Aalborg University Copenhagen, Denmark)  
Federico Avanzini (University of Milano, Italy)

## **Summer School Chair**

Romain Michon (Stanford University, USA)

## **Music Chairs**

Antonio Jurado (Universidad de Málaga, Spain)  
Juraj Kojš (Slovakia/USA)  
Spencer Topel (Dartmouth College, USA)

## **Demo Chair**

Ana M. Barbancho (Universidad de Málaga, Spain)

## **Local Committee**

Alberto Peinado (Universidad de Málaga, Spain)  
Alejandro Villena (Universidad de Málaga, Spain)  
Irene Gómez (Universidad de Málaga, Spain)  
Jose María Gómez Belmonte (Universidad de Málaga, Spain)  
M<sup>a</sup> Luna Herruzo Torrico (Universidad de Málaga, Spain)  
Juan Ignacio García Bartolomé (Universidad de Málaga, Spain)  
Moisés Marín Amador (Universidad de Málaga, Spain)

**Program Committee**

Areti Andreopoulou (National and Kapodistrian University of Athens)

Anastasia Georgaki (National and Kapodistrian University of Athens)

Stefan Bilbao (University of Edinburgh)

Jean Bresson (Institut de Recherche et Coordination Acoustique / Musique)

Emma Frid (Royal Institute of Technology Stockholm)

Lauren Hayes (Arizona State University)

Alexander Refsum Jensenius (University of Oslo)

Luca Andrea Ludovico (University of Milano)

Andrew McPherson (Queen Mary University London)

Romain Michon (GRAME Centre National de Création Musicale)

Laurel Pardue (Queen Mary University London)

Sandra Pauletto (University of York)

Bob Sturm (Royal Institute of Technology Stockholm)

Hiroko Terasawa (University of Tsukuba)

Andrea Valle (University of Torino)

---

# SMC 2019 Contributions





# DAW-Integrated Beat Tracking for Music Production

**Brett Dalton**  
University of Victoria  
brettdalton.ca@gmail.com

**David Johnson**  
University of Victoria  
davidjo@uvic.ca

**George Tzanetakis**  
University of Victoria  
gtzan@uvic.ca

## ABSTRACT

Rhythm analysis is a well researched area in music information retrieval that has many useful applications in music production. In particular, it can be used to synchronize the tempo of audio recordings with a digital audio workstation (DAW). Conventionally this is done by stretching recordings over time, however, this can introduce artifacts and alter the rhythmic characteristics of the audio. Instead, this research explores how rhythm analysis can be used to do the reverse by synchronizing a DAW's tempo to a source recording. Drawing on research by Percival and Tzanetakis, a simple beat extraction algorithm was developed and integrated with the Renoise DAW. The results of this experiment show that, using user input from a DAW, even a simple algorithm can perform on par with popular packages for rhythm analysis such as BeatRoot, IBT, and aubio.

## 1. INTRODUCTION

Tempo is a feature of audio which is commonly analyzed in Music Information Retrieval (MIR) due to its fundamental role in music. Tempo is described by a pulse, a set of steady intervals of time which govern the way that music is perceived and expressed. In a written piece of music, notes and rhythmic events are aligned in relation to this pulse, which is grouped and subdivided in various ways, ultimately forming the piece's structure. This underlying structure is necessary for musicians to synchronize with each other while performing, and it allows listeners to understand what they hear. The importance of tempo in music makes for a broad range of practical applications for its analysis, ranging from genre classification to DJ software, [1], [2], [3].

In digital music production, tempo is most often described by a piece's global beat rate, commonly measured in Beats Per Minute (BPM). Generally this is understood as a constant value which does not vary over the duration of the piece. This is different from how music is traditionally performed by live musicians, where the tempo will naturally drift without mechanically perfect synchronization to some clock.

This presents three possibilities for digital music producers trying to work with live source material: One, to synchronize the recording with a constant BPM by using time-

stretching; two, to extract the structure from the original piece and synchronize their software to the recording; or, three, to work with the musical structure by ear, without digital assistance.

In order to explore the second option, this paper uses the work of Percival and Tzanetakis [4] with some modifications and additions, to demonstrate the effectiveness of beat tracking in a modern music production setting. This has been accomplished by developing a piece of software which integrates a simple beat extraction algorithm in a Digital Audio Workstation (DAW) to accomplish tasks which would otherwise be cumbersome when working with live recorded material.

The DAW that has been chosen for this research is Renoise, as it has a native scripting API that can be used to modulate tempo over time. The core algorithm was written and tested in Python, and Lua was used to integrate the code with Renoise's scripting interface. Later, the algorithm was ported to C for real-time use. The results of this work have been evaluated using mir-eval<sup>1</sup> [5], a python library designed for gathering statistics for common MIR tasks. The results are compared to the established beat tracking methods aubio [6], IBT [7], and BeatRoot, [8].

## 2. BACKGROUND

Beat tracking is a well researched area in MIR that has been analyzed in different contexts using a variety of methods. Early approaches to beat tracking involved processing the symbolic representations of music such as musical scores and MIDI data. However, as computing technology and theory developed, it became possible to analyze raw audio recordings for the purpose of beat extraction and the extraction of more complex metric information, [9], [10].

As noted by Gkiokas et al., most tempo and beat extraction algorithms share some common structural elements, [11]. The most common approach for processing an audio signal involves retrieving what is known as an onset strength signal (OSS), novelty curve, or salience function, [2], [12], [13]. This is a transformation of the original audio that attempts to capture a continuous function of rhythmic importance over time. A number of different metrics can be used to derive the OSS such as spectral flux, phase deviation, and complex domain methods, [14] as well as machine learning [15]. Peaks in this signal can be thought of as discrete rhythmic events that can be used in further analysis to extract information using a variety of techniques. These techniques may involve multi-agent

Copyright: © 2019 Brett Dalton et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> [https://github.com/craffel/mir\\_eval](https://github.com/craffel/mir_eval)

systems [16], autocorrelation [4], and it is also common to use higher level musical features such as chord changes and drum patterns to obtain more accurate results, [10].

As of today, few DAWs incorporate true beat tracking for recorded material, but instead offer tools for manually solving tempo related problems such as audio to MIDI conversion, BPM detection, time stretching, and quantization. This is perhaps due to beat tracking being less reliable and less flexible than the alternatives.

There has been past work in creating an interactive system for beat tracking. In 2001, Dixon developed graphical software for beat extraction, data editing, sonification, and a variety of other tasks, [17]. At the time, he noted some deficiencies in similar tools; one being that they did not allow for the user to correct mistakes in the beat tracking process. The goal of creating this beat tracking software is to continue on Dixon's line of work and integrate a graphical user interface with a beat tracking system that is useful for real world applications in a music production environment.

### 3. ALGORITHM DESCRIPTION

Our proposed beat extraction algorithm contains the following stages similar to other methods: extract an onset strength signal from the desired audio file; extract individual onsets and their attributes such as loudness and timing with peak picking; and iterate over these onsets to find potential beats using a simple heuristic induction method.

The Streamlined tempo induction algorithm developed by Percival and Tzanetakis is designed with the intention of using the simplified forms of common techniques while still attaining reliable performance [4]. The task of tempo induction has different requirements from beat extraction, but they are naturally related. Our algorithm employs the same OSS and peak picking methods as the Streamlined algorithm, with some modifications and additions.

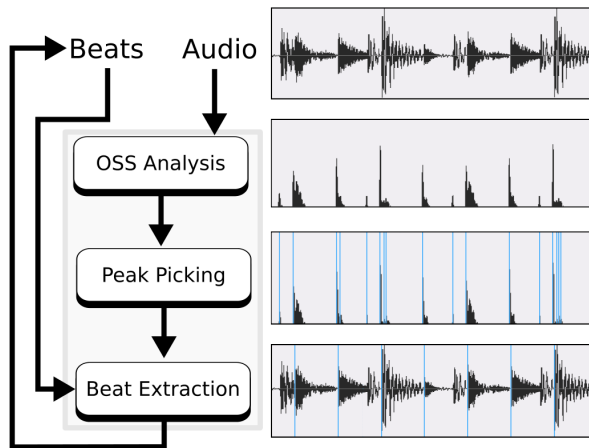


Figure 1. The proposed block diagram of the beat tracking system. Note the ability to generate, modify, and re-process beats.

### 3.1 OSS Calculation

OSS is normally calculated using the spectral flux of the input signal. This involves taking a Short Time Fourier Transform (STFT) and taking the sum of all frequencies bins which increase in energy from frame to frame. We ignore the decreasing bins because they are typically less rhythmically relevant.

$flux(t)$  is the raw spectral flux function.  $rect(x)$  is the rectification function, such that  $rect(x) = 0$  where  $x < 0$ , elsewhere  $rect(x) = x$ .  $X$  denotes the Fourier transform of a signal as a function of time and frequency.

$$flux(t) = \sum_{i=0}^N rect(|X(t, i)| - |X(t-1, i)|) \quad (1)$$

Our OSS calculations are a slight modification of the Streamlined algorithm. The original approach involved using a low-passed copy of the result of the flux calculations to remove noise, but we use the unfiltered flux to preserve the exact timing of fluctuations, something more valuable for beat extraction than for global tempo analysis.

This approach gives a decent metric of where rhythmically salient events occur, however, it is flawed because it gives much less weight to low frequency fluctuations. By definition, low frequency signals fluctuate more slowly than high frequencies, and, as such, their fluctuations will have a much lower amplitude within a given STFT window. This means that this method is not able to clearly distinguish between a short click and bass drum, despite the fact that the bass drum is much more important for rhythmic perception.

In an attempt to remedy this, we make a copy of the flux using a downsampled copy of the input, and then mix it into the final output. We'll call this  $flux_{adj}(t)$ , as described below, with  $a = 0.85$  and  $b = 0.15$ .

$$flux_{adj}(t) = aflux(t) + bflux_{downsampled}(t) \quad (2)$$

Overall, this minor addition marginally improves results, but a better solution is worth investigating. Going forward, it may be worthwhile to use a multiband OSS calculation, as described by Bock, [18], or develop a different spectral flux calculation that compensates for bin fluctuation over frequency.

### 3.2 Peak Picking

Our next step is to extract each onset event with peak picking. We high-pass the OSS, which produces a new signal where sudden increases in flux are positive, and sudden decreases are negative. We segment the signal based on the positive zero crossings and find the maximal value for each region. We add these local maxima times to the set of peaks which gives us a set of discrete events that we can use for our final prediction.

The following expression defines the set of all positive high-passed OSS zero crossings:

$$\{cross_i = t \mid OSS_{hp}(t) > 0 \text{ and } OSS_{hp}(t-1) < 0\} \quad (3)$$

$peak_i$  refers to the position of each onset event in time. The  $\text{argmax}$  function uses the last two parameters to specify a range.

$$peak_i = \text{argmax}(OSS_{hp}, cross_i, cross_{i+1}) \quad (4)$$

### 3.3 Beat Extraction

Next we must determine which beats to select. In the case that we have an estimate of the time delta between beats, we can use the error as a simple metric for determining if the peak is a beat. We get this expected beat time through induction by using a previous known beat's time plus some expected delta. Two initial beats are provided as input to the algorithm to begin the induction process. In our system, these are created by the user. Finding the peak with the minimum error is a trivial procedure on an ordered list of peaks; The error will increase monotonically, so once we find a peak with a positive error, no subsequent peaks will have a smaller error. This is the basic strategy for building our set of beats.

$\varepsilon(i, j)$  is the error for a given peak event,  $i$ , with the prior beat,  $j$ . If no peak is found within a certain error threshold, we skip a beat to find the next one.

$$\varepsilon(i, j) = (peak_i - beat_{j-1}) - \delta_\mu(j) \quad (5)$$

In order to allow for varying tempos, we define a local beat delta,  $\delta_\mu(i)$ , which is expressed in terms of previously found beat deltas.  $\delta_\mu(i)$  describes the expected beat delta for  $beat_i$  as a function of previous beat deltas. Larger values of the averaging window,  $N$ , will result in more stable BPM fluctuations.

$$\delta_\mu(i) = \frac{\sum_{j=1}^N \delta_{i-j}}{N} \quad (6)$$

Finally we output this set of beats, which we will use to generate a tempo curve of the entire audio recording. An initial beat delta can either be supplied manually or predicted by an algorithm, such as the Streamlined algorithm. We also define an error tolerance, such that if no beat is found within that tolerance we double the length of the beat time and continue to search. When we do find a valid peak, we fill in the missing beats by subdividing the beat and reset the expected beat delta. Some optional parameters include the OSS threshold level, error tolerance and the averaging window,  $N$ , for the expected beat delta.

## 4. DAW INTEGRATION

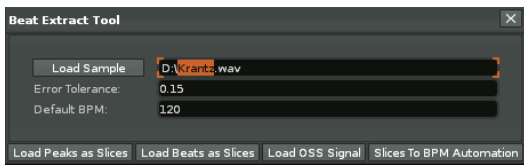


Figure 2. An image of the beat extraction tool made with the Renoise scripting API.

The Renoise DAW provides a scripting API that allows users to develop their own tools using the Lua scripting language. The API allows the developer to access some of the DAW's internal GUI elements and automate certain production tasks.

The tool that has been created to integrate this algorithm with Renoise takes advantage of Renoise's sample editor, which already has the ability to place slice markers that partition an audio file into multiple segments. The algorithm takes a sequence of beat times as inputs and generates a sequence as output.

To use the tool, the user can add two or more markers to initialize the algorithm, and then they can press the "Load Beats as Slices" button, which will run the algorithm on this input, generating a new set of slices. These generated slices can then be edited further, or they can be used to generate a tempo curve with the "Slices to BPM Automation" button. Extra beats can be added at various places in the audio file to correct errors, and ensure that the system correctly tracks erratic variations in tempo. These features constitute a user-in-the-loop interface, where the user can see the results of the algorithm, tweak the inputs, and then generate more refined output, as seen in figure 1.



Figure 3. Renoise's sample editor. The orange lines are slice markers which the user can use to supply initial beat info.

There are a number of parameters that can be exposed to the user to fine tune the behaviour of the algorithm, however, they have not yet been integrated into the tool. These parameters include the confidence threshold, peak detection threshold, beat error tolerance and so on. This would make it easier to work with different kinds of material, as inputs that have more erratic rhythmic fluctuations will require more fine tuned parameters to produce accurate results.

The tool was tested with various recordings by the author, and it was found to work very well with music featuring strong percussion such as rock and jazz. However it was not suitable for use with less percussive recordings such as piano music. It may be useful in the future to study the effectiveness of the tool by comparing the time differences between manual and automatic beat annotation in a user study. This would give some more objective grounds to decide whether the algorithm is ready for practical use.

## 5. EVALUATION

The mir-eval<sup>2</sup> [5] evaluation suite was used to evaluate the efficacy of the algorithm against the beat tracking algorithms in the aubio [6], IBT [7], and BeatRoot [8] libraries.

<sup>2</sup>[https://github.com/craffel/mir\\_eval](https://github.com/craffel/mir_eval)

Algorithm	Aubio [6]	IBT [7]	BeatRoot [8]	Proposed (automatic)	Proposed (user initialized)
F-Measure	0.57	0.27	0.70	0.50	0.95

Table 1. F-Measure results from mir-eval. In the automatic column, the proposed algorithm is seeded with a bpm from the Streamlined tempo estimation algorithm, whereas in the user initialized column the algorithm is given two beats from the ground truth.

There are some complications when comparing this algorithm with other methods due to the fact that user input is used for initial BPM estimations, whereas other methods predict the BPM automatically. This puts the other algorithms at a disadvantage, and so it is not a fair comparison. Of the three methods tested, it was unclear if the host framework (Sonic Annotator) [19] provided the ability to supply a BPM estimate or initial beat. In order to show the difference in performance between the automatic and human-in-the-loop approach, the experimental results provided show our algorithm using user-supplied initial beat inputs (user initialized) as well as automatically generated input using the Streamlined tempo estimation algorithm (automatic). As seen in table 1, the algorithm performs much better when supplied with initial beats, bringing its F-Measure of 0.50 up to 0.95.

The user supplied beat method uses two beats from the ground truth for each file in order to initialize the algorithm with a valid initial beat and initial beat delta. One concern with this approach is that mir-eval's F-measure score includes these shared beats, but their contribution to the score is negligible because they only make about 40 of the 1300 beats in the entire dataset. Since the ground truth annotations were derived from listeners tapping along to the audio, this should be a fair approximation of a real human-in-the-loop use case.

The data used includes 19 files from Sound and Music Computation for MIREX 2017, [20]. The set of files used were ones labelled as being easy; the files named SMC\_271 through SMC\_289. Each audio clip in the dataset is 40 seconds long and contains roughly 30 to 130 beats depending on the tempo. These files have strong percussive elements, while the remainder of the dataset is mostly classical music with weak rhythmic features. As stated previously the OSS calculation used is not adequate for detecting In the future this test data will need to be expanded, and the algorithm should be improved to work with less percussive audio sources.

The takeaway from these results is that a small amount of user provided information can be used to significantly improve the performance of a simple beat tracking algorithm. Other methods have been able to achieve 60%-80% F-measure accuracy, [21], [22], [12], and some methods that have been able to achieve up to 90% accuracy, [15]. This shows that the reframing of the beat induction problem allows a simplified algorithm to perform on par with, and even surpass, some of the most sophisticated algorithms within a limited practical context. These results are encouraging for further research into how these methods can be used to improve the functionality and usability of DAW software.

## 6. APPLICATIONS & FURTHER WORK

In this paper we have explored one of the more immediately obvious uses of beat tracking, which is synchronizing DAW software to an audio recording. There is, however, a wealth of other potential applications for beat tracking ranging from practical to experimental in nature.

Any given set of beats yields a tempo curve which is the function of tempo over time. The tempo curve is a signal like any other, and can be manipulated using filters and other conventional signal processing techniques. At the macro scale, the filtering of tempo curves can be used to remove tempo drift from recordings, as well as quantizing and performing other rhythmic corrections. With a high resolution tempo curve that captures rhythmic variation below the beat level, it would be possible to extract and manipulate musical rhythms in even more novel ways. One could imagine using a "tempo equalizer" on the tempo curve of a jazz recording to increase the amount of swing, or remove it entirely. The extraction of tempo curve information combined with time stretching and other audio manipulation techniques has many promising applications, and yet few, if any, of these applications have been realized in common audio software.

## 7. CONCLUSION

The results of this work show a promising potential for beat tracking algorithms in Digital Audio Workstations. We showed how simple beat tracking methods can be used to reliably synchronize DAW tempo with an audio source with user input and interactive corrections.

Currently the most significant concern with the method used in this work is the limited data used for testing. For future work we plan to conduct experiments with a larger and more varied data set. The algorithm also needs to be improved in order to deal with music that does not have strong percussive elements. Therefore, to make this viable, the most urgent issue to be addressed is the OSS calculation. The development of a new metric which is more able to capture rhythmic salience will be required for further work with material that does not contain strong transients.

Nevertheless, our method produces accurate results when provided with a small amount of user input, even surpassing the performance of conventional methods. This method is convenient and easy to use for production purposes in a DAW, and could easily be augmented with more sophisticated beat tracking techniques. With further work, DAW integrated beat tracking may become a powerful and interactive tool for music producers looking for creative ways to manipulate rhythm in digital music.



## 8. REFERENCES

- [1] K. Jensen and T. H. Andersen, "Beat estimation on the beat," in *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on*. IEEE, 2003, pp. 87–90.
- [2] D. P. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.
- [3] J. L. Oliveira, M. E. Davies, F. Gouyon, and L. P. Reis, "Beat tracking for multiple applications: A multi-agent system architecture with state recovery," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 10, pp. 2696–2706, 2012.
- [4] G. Percival and G. Tzanetakis, "Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 12, pp. 1765–1776, 2014.
- [5] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "mir\_eval: A transparent implementation of common mir metrics," in *In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR*. Citeseer, 2014.
- [6] P. Brossier, "Automatic annotation of musical audio for interactive systems," Ph.D. dissertation, Ph. D. thesis, Queen Mary University of London, London, UK, 2006.
- [7] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.
- [8] C. Cannam, M. Mauch, M. E. P. Davies, S. Dixon, C. Landone, K. C. Noland, M. Levy, M. Zandoni, D. Stowell, and L. A. Figueira, "Mirex 2013 entry: Vamp plugins from the centre for digital music," 2013.
- [9] M. E. Davies and M. D. Plumbley, "Context-dependent beat tracking of musical audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 3, pp. 1009–1020, 2007.
- [10] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *Journal of New Music Research*, vol. 30, no. 2, pp. 159–171, 2001.
- [11] A. Gkiokas, V. Katsouros, G. Carayannis, and T. Stafylakis, "Music tempo estimation and beat tracking by applying source separation and metrical relations," in *ICASSP*, 2012, pp. 421–424.
- [12] P. Grosche, M. Müller, and C. S. Sapp, "What makes beat tracking difficult? a case study on chopin mazurkas," in *ISMIR*, 2010, pp. 649–654.
- [13] S. Dixon and E. Cambouropoulos, "Beat tracking with musical knowledge," in *ECAI*, 2000, pp. 626–630.
- [14] S. Dixon, "Onset detection revisited," in *Proceedings of the 9th International Conference on Digital Audio Effects*, vol. 120. Citeseer, 2006, pp. 133–137.
- [15] —, "Learning to detect onsets of acoustic piano tones," in *Proceedings of the Workshop on Current Directions in Computer Music Research*, 2001, pp. 147–151.
- [16] M. Goto and Y. Muraoka, "A real-time beat tracking system for audio signals," in *ICMC*, 1995.
- [17] S. Dixon, "An interactive beat tracking and visualisation system," in *ICMC*. Citeseer, 2001.
- [18] S. Böck, "Event detection in musical audio," Ph.D. dissertation, PhD thesis. Johannes Kepler University Linz, Linz Austria, 2016.
- [19] C. Cannam, M. O. Jewell, C. Rhodes, M. Sandler, and M. d'Inverno, "Linked data and you: Bringing music research software into the semantic web," *Journal of New Music Research*, vol. 39, no. 4, pp. 313–325, 2010.
- [20] A. Holzapfel, M. E. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon, "Selective sampling for beat tracking evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 9, pp. 2539–2548, 2012.
- [21] A. Pikrakis, I. Antonopoulos, and S. Theodoridis, "Music meter and tempo tracking from raw polyphonic audio," in *ISMIR*, 2004.
- [22] A. Mottaghi, K. Behdin, A. Esmaeili, M. Heydari, and F. Marvasti, "Obtain: Real-time beat tracking in audio signals," *arXiv preprint arXiv:1704.02216*, 2017.

# INTERACTION-BASED ANALYSIS OF FREELY IMPROVISED MUSIC

Stefano Kalonaris

Center for Advanced Intelligence Project (AIP)

RIKEN, Japan

stefano.kalonaris@riken.jp

## ABSTRACT

This paper proposes a computational method for the analysis and visualization of structure in freely improvised musical pieces, based on source separation and interaction patterns. A minimal set of descriptive axes is used for eliciting interaction modes, regions and transitions. To this end, a suitable unsupervised segmentation model is selected based on the author's ground truth, and is used to compute and compare event boundaries of the individual audio sources. While still at a prototypal stage of development, this method offers useful insights for evaluating a musical expression that lacks formal rules and protocols, including musical functions (e.g., accompaniment, solo, etc.) and form (e.g., verse, chorus, etc.).

## 1. INTRODUCTION

When tackling musical structure, it is not uncommon to employ language-based conceptual blends<sup>1</sup>. Some of these focus on a formal and generative grammar approach, while others foreground metapragmatics and conversational metaphors. While the former [2] are conditioned upon a notion of *musical surface*<sup>2</sup> and focus on hierarchical structures of musical phenomena in the context of Western tonal music, the latter have been employed for less formal theories, when dealing with musical improvisation practices, such as *jazz*.

A different perspective is needed when analyzing *free jazz* [3,4] or *free improvisation* [5,6], which are musical expressions that lack an agreed upon representation scheme, and which defy and challenge definitions and categorizations. While recent work has been done in this field to understand how structure is perceived in these musical expressions [7], more research is needed in this regard.

This paper foregrounds the dialogical component of this music, whereby structures are negotiated in real-time, emerge ad-hoc, and cannot be inferred or deduced from a score. A method for the structural segmentation and analysis of musical improvisations of this kind is proposed, in-

spired by Pelz-Sherman's [8] speculations on interactional listening/music making.

Conditioned upon the analysis of the individual voices (audio sources), this study considers a multi-track recording and implements a distilled version of Pelz-Sherman's scheme, whereby interaction patterns and dynamics are deduced comparing individual boundaries and audio features spaces. By doing so, the method offers itself as a tool for investigation of how structure, in this context, might emerge from the continuous negotiation of musical expectations and demands, how these might be communicated to others, acknowledged or ignored altogether.

## 2. CONTEXT

Free jazz and free improvisation are not the same musical expression. While the latter is often viewed as the avant-garde European offshoot of the former, they are distinct expressions which can be easily discriminated. However, they also share sufficiently many characteristics, from ideological to musical. For example, they both share the desire to rebel against the *status quo*, to assert freedom from conventions and uniformity, not without political and societal implications.

Generally speaking, no predefined agreement or commitment about the music is made and, according to this paradigm, players negotiate the musical outcome in real-time. In free improvisation it is customary not to abide by musical referents (such as idiom, style, genre, or even tonal keys), while free jazz has a stronger element of idiomatic playing, linked to the broader development and narrative of Afro-American musical expression. Of course, and despite occasional claims of the contrary, there is no such thing as an unbound, *ex nihilo* improvisation since all musicians have an acquired protocol of interaction, based on historical, personal or shared aesthetic and musical preferences. Despite this, free jazz and free improvisation are arguably less formalized than other improvised expressions (e.g., a cadenza in a solo concerto). Paramount to both is the focus on interaction, distributed decision making and lack of predefined musical outcome. Furthermore, "improvisation must be open - that is, open to input, open to contingency - a real-time and (often enough) a real-world mode of production" [9, p.38]. For the sake of simplicity free jazz and free improvisation will be hereinafter referred to as *freely improvised music*, without specific preference for one or the other, unless explicitly stated.

To attempt a computational analysis of this musical expression it is crucial to work from the individual musical

<sup>1</sup> An integration procedure formalized by Fauconnier [1].

<sup>2</sup> A discrete representation of the sounds in a piece, to include pitches, durations and dynamics, intrinsically linked to the concept of music notation.

parts, to see how they relate to one another and how these relations evolve over the course of the piece.

## 2.1 Source Separation

Attempts to source separate historic and representative freely improvised recordings were made by the author, using non-negative matrix factorization with the Flexible Audio Source Separation Toolbox (FASST) [10] and harmonic-percussive separation [11]. However, results in this respect were not deemed satisfactory and, given that the state of the art in source separation was not the principal focus of this study, improvements on this front were not pursued further. Instead, a multi-track recording from the MedleyDB [12] (see Section 4.2) was used, for lack of a better alternative and despite stronger idiomatic assumptions. Source separation is an open problem and an active topic of research, and it has never, to the author's knowledge, been applied to freely improvised music (although it has been explored for early jazz recordings [13]). This might be due to the low commercial and aesthetic appeal and popularity of this musical expression. Source separation thus far has been more concerned with application to the music industry and to creative music technologies, e.g. automatic mixing [14], automatic transcription [15], orchestration [16] or voice separation [17]. In this context, clearly defined musical functions and registers for the instruments are preferred: drums play rhythm, harmonic instruments play chords, melodic instruments play melody, vocals float on top, and so forth.

A scenario of this type is undesirable in freely improvised music. In fact, such compliance with predefined roles and domains is the primary impetus out of which free jazz and free improvisation were developed in the first place. In these musical expressions it is common practice to use extended techniques, whereby the spectral palette of each stream is augmented beyond "normal", shifting the attention from melodic contour to gestural morphologies of sound, such as trajectories, density, functional relations and so forth. The spectral spill-over generated by this mode of playing, along with non exclusive musical roles (e.g., a guitar can be hit with mallets and objects and used as a percussion, etc.) makes it challenging to clearly separate the sources.

## 2.2 Structure in Freely Improvised Music

The issues linked to spectral spill-over and musical role cross-over go beyond source separation tasks, and can also make the use of standard music information retrieval (MIR) techniques for structural segmentation arduous. Notions commonly used in MIR tasks and cognitive-based approaches to musical surface parsing and segmentation rely on culture-specific axioms. For example, the predicate that major and minor triads form convex subsets [18] (which had already been challenged by Forte's music set theory [19]). Assumptions made in this contexts are difficult to port to musical domains that do not share the same tonal/functional axioms.

Adding to the the difficulties in deducing musical structure in freely improvised music is the issue of represen-

tation. Roads [20] posits that one can represent music at three levels: iconic, symbolic and score level. The first would include data relative to an audio waveform (e.g., sequences of values for amplitude and phase) or graphic scores, the second would include the use of signs which would convey syntactical meaning, and the third can be assimilated to what is commonly called music notation. No attempt to define a representation system or scheme for free improvisation has been made up to date, although graphic scores or snippets of musical notation [21] can be used as platforms (sometimes distributed, collaborative and editable on-the-fly [22,23]) for inspiration and suggestive/aleatoric interpretation, especially in free jazz. Nevertheless, these tools cannot fully describe and contain the musical process and product. Audio representations can also be used, but retrospectively. That is, the music is always created on the spot and no musician knows what the outcome will be *a priori*.

The issues outlined above, however, do not imply that these musical expressions lack structure or structural segments [24, 25]. More specifically, it appears that macrostructure in freely improvised music is a surface phenomenon emerging from micro-structures which are sufficiently differentiated at some feature level. The transitional regions between these sections are paramount for the understanding of segmentation boundaries and of their treatment (e.g., gradual, clear-cut, etc.) in real-time. It has been shown that expert improvisers can "generate segmentation in high-level musical structure" [24, p. 235].

## 3. AN INTERACTION-BASED VIEWPOINT

Improvised music can be challenging in many ways, but can be better understood as a dynamical and distributed decision making process. Structure is thus a by-product of such process, which has been investigated in terms of saliency and coordination [26]. According to this interpretation, the notion of *focal point* is paramount. This can be defined as "a point of convergence for expectations" [26, p.3].

As seen so far, negotiation, coordination and interaction are the key concepts needed to understand and analyze freely improvised music. The focus of this paper is in fact on the real-time interaction of potentially several musical "voices", and the relational nature of the musical result. To this end, it is useful to think of a relationship as the association between two elements/agents which emerges via a specific connection, and of interaction as the events and actions that help (or not) to form or define a relationship<sup>3</sup>.

Having established this relational/interactional paradigm, some analogies and metaphors naturally come to the fore, such as the parallels between music-language and improvisation-conversation. While these pairs seem related (two people need a common language to have a conversation, just as musicians might need to operate within given conventions to musically communicate), the correspondences are subtle, contradictory and often problematic.

<sup>3</sup> For example, an estranged father has a filial relationship with his offspring and a null interaction.



### 3.1 Linguistic Approaches

Theories inspired by Chomsky’s generative grammars [2, 27–30] are common in the context of Western tonal music, and in computational musicology. They have also been proposed for musical improvisation [31], with limited application. Grammar-based perspectives of musical representation fall under the symbolic category, whereby deep structures are inferred, parsed or deduced from the musical surface. These approaches have not been unchallenged. Some have doubted the ability of linguistic-based and Gestalt-based approaches to generalise across listeners at a macro-structural level [32], others argue for a more holistic perspective of sounds [33], others yet note that high-level entities, like beat structure<sup>4</sup>, chord simultaneities<sup>5</sup> and voice separation<sup>6</sup> are necessary for the formation of the musical surface [37]. Importantly, freely improvised music has no (or very little) dependency on musical surface, as pieces are not planned, notated and performed accordingly, as discussed in Section 2.2 (although retrospective notation by means of transcription can be done).

Furthermore, a grammar-based representation of musical deep structure is purely functional, and fails to account for sociological, emotional, moral, aesthetic, and cultural aspects involved in musical expression. The main difficulty encountered when employing grammar-based approaches to music representation is that of modelling context. This is particularly problematic in improvised music, where phrasing and context are often interrupted and re-instantiated. To this end, Roads [20] suggests that more research should be undertaken in “interrupt-driven” grammars.

### 3.2 Conversational Approaches

Musical interaction occurring in improvisation has been often viewed under the paradigm of verbal communication. Drawing from *metapragmatics* [38], both Sawyer [39] and Monson [40] develop their frameworks for understanding jazz improvisation through a conversational metaphor. Sawyer, in the larger context of improvisational studies, notes that “improvisational interaction can be mediated by both linguistic and musical symbols” [41, p. 150]. Improvisation is thus associated to different voices in dialogue with one another, a real-time conversation (without a pre-defined topic). It is arguable that a successful conversation relies on effective communication. While more idiomatic forms of musical improvisation such as *jazz* focus more on narrative [42] and story telling [43], actively engaging with tradition and lineage, freely improvised music is less preoccupied with linear accounts and more focused on real-time distribution of agency and the dialogical aspect of communication [44].

<sup>4</sup> Beat structure comprises *beat induction* (finding an appropriate relative clock) and *beat tracking* (a dynamically changing clock).

<sup>5</sup> Chord simultaneity presupposes culture-specific knowledge stored in long-term memory [34] and it is an *emergent quality* [35].

<sup>6</sup> The auditory system can decompose spectral fusions [35, p.64] into separate streams, based on pattern analysis. This process was described in [36] as “auditory stream segregation”.

### 3.3 Interactional Music-making

Despite the many similarities between music and language or musical improvisation and conversation, there remain sufficiently many fundamental differences to warrant caution when blending domains. Beyond the inadequacy of formal grammars or the lack of formal theories in conversational approaches, the relational nature of the musical interactions occurring in freely improvised music is paramount, and it is the principal motivation for the method proposed in this paper. A dialogical perspective foregrounds such interaction between the musical constituent parts, which can be assimilated to the audio sources (streams) in a recording. Stream segregation with respect to music improvisation has been investigated in [45], where the concept of interactional listening is developed. Interaction is also the pivot of Peltz-Sherman’s work, which is reviewed in the next section to investigate boundary localization and segmentation of freely improvised musical pieces.

## 4. METHOD

### 4.1 Overview

According to Pelz-Sherman’s [46] distinction between monoriginal and heteroriginal musical expressions, freely improvised music classes among the latter. He posits [8] that performers are, at any given time, either in a state of transmitting and/or receiving musical signals, with *i-events* representing the mutual response to a musical request, called a *cue*. Pelz-Sherman does not offer an exhaustive list of *i-events*, but he lists *imitation*, *question and answer*, *completion/punctuation* and *interruption*. Furthermore, he discriminates between static and dynamic modes of interaction, whereby the former (*sharing*, *solo/accompaniment*, *not sharing*) are fundamental states at which players operate at any given time, and are associated with levels of *i-event* density (high, medium, low, respectively). Dynamic modes (*emerging/withdrawing*, *merging/accepting*, *interjecting/supporting*, *initiating/responding*) instead can be thought of as the types of transitions between any two static modes. To further clarify, static modes of interaction can be assimilated to the inter-boundary regions in the context of structural segmentation, whereas dynamic modes can be considered the intra-boundary segments. Other frameworks have been proposed in this context, some of which have a more extensive taxonomy of transitions and/or relational functions [47], however, these were not considered in the current study for the sake of simplicity.

In this paper Pelz-Sherman’s scheme is reduced to comprise two essential descriptors: *static mode* (spanning from not sharing to sharing) and *dynamic mode* (either morphing or clear-cut). Concretely, the former is a measure of similarity between the musical features of the audio sources in-between the segment boundaries, whereas the latter is a level of agreement between the boundary placement/detection over them. To this end, the author posits that if a boundary is detected in most parts in a given time window (thus, a cue was responded to within this threshold), then a (more or less) clear-cut transition is assumed.

Algorithm	Accuracy	Precision	Recall	F1 score
vmo (mfcc)	0.805	0.182	0.133	0.154
foote (mfcc)	0.8	0.2	0.056	0.088
cnmf (mfcc)	0.8	0.2	0.056	0.088
olda (mfcc)	0.81	0.375	0.167	0.231
scluster (mfcc)	0.762	0.182	0.111	0.138
sf (mfcc)	0.79	0.167	0.056	0.084

Table 1: Metrics for the different (MFCC-based) algorithms used for segmentation

Conversely, if inter-part segment boundaries do not agree, a morphing transition is assumed. This might be a scenario whereby one player sends a cue which is not followed by a significant change in the musical feature space of the other player(s) (either as the result of a deliberate musical choice/strategy, or simply because they missed it).

## 4.2 Procedure

To test the method, the multi-track recording *FreeJazz* by MusicDelta, from the MedleyDB [12] was used. This trio (clarinet, double bass and drums) recording was chosen for several reasons. Firstly, and as discussed in Section 2, free jazz shares many of the broader concerns of freely improvised music (such as the desire to break regular tempos, tones, and chord changes conventions). Secondly, the track was deemed by the author, a domain expert and practitioner, sufficiently apt to investigate multi-part interaction. Thirdly, and after extensive search, it was not possible to source an historical example of either free jazz or free improvisation in multi-track format. Several constraints (e.g., time, location, musicians’ network) at the time of writing did not allow for a bespoke recording of a multi-track piece.

The raw individual audio sources and the audio mix were segmented using the MSAF Python package [48], based on several of the available algorithms (e.g., variable Markov oracle [49], audio novelty [50], convex non-negative matrix factorization [51], ordinal linear discriminant analysis [52], spectral clustering [53]), each in turn based either on Mel-frequency cepstrum coefficients<sup>7</sup> (MFCCs) or tempogram features. The former are shown in Figure 1, for comparison’s sake.

To choose one of these algorithms, the results on the audio mix were compared to the author’s analysis of the same file. This analysis was used as the ground truth for computing the F-score, shown in Table 1.

The ordinal linear discriminant analysis (OLDA) [52] algorithm was selected, based on its score. In Figure 2 the ground truth and the detected bounds are plotted for comparison. Using the best performing model, the boundaries obtained on the audio sources were compared. Figure 3 illustrates these boundaries, and reveals several salient events in the piece. Saliency is inferred because the boundaries feature in both the audio mix and the individual sources.

<sup>7</sup> Relating to a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear scale of frequency.

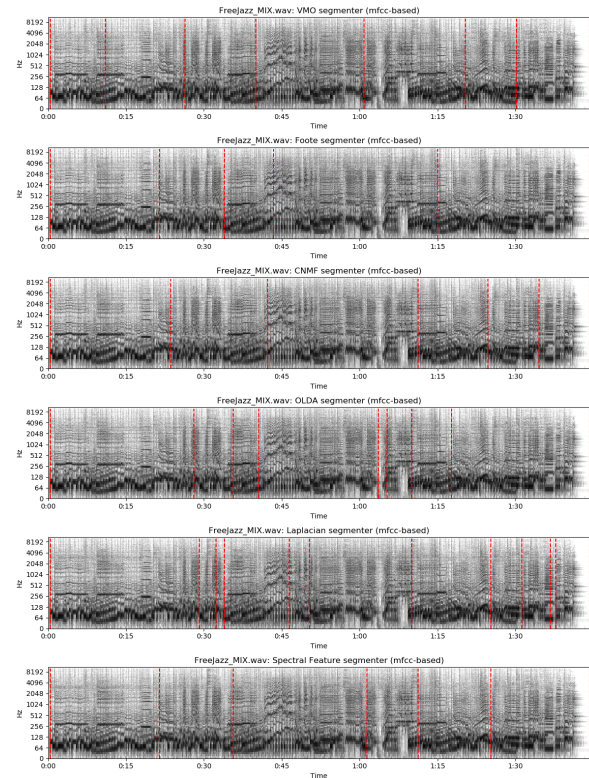


Figure 1: Segmentation boundaries on the audio mix, using several algorithms and based on MFCC features.

Based on this consideration, clear-cut and morphing transitions were identified, according to whether the individual audio sources’ boundaries agreed (+ or - a 2 seconds<sup>8</sup> buffer, factored in to account for the reaction time needed by one player to respond to a musical cue originated from the other players) or not, respectively. This procedure formalizes what was called dynamic mode in Section 4.1 and it is shown in Figure 4.

Static mode, on the other hand, is concerned with the sharing of the musical feature space at a given time. To this end, regions in between the boundaries were used to compute the similarity over given audio features. Figure 5 shows the inter-regions zero-crossing rate<sup>9</sup> dynamics for all three audio sources. A total of 27 such features were initially computed and are available for inspection, although they are omitted here for the sake of brevity.

In the example shown in Figure 6, 6 features (root-mean-square energy, spectral centroid, spectral bandwidth, spectral flatness, spectral roll-off, and zero crossing rate) were used to calculate the (average) inter-region similarity between the audio sources, using cosine similarity (Pearson correlation or other metrics are also possible). The similarity values so obtained were used as the color gradient.

<sup>8</sup> This time window is heuristically determined, and used to divide the audio buffer into bins of this length.

<sup>9</sup> The rate at which the signal changes from positive to zero to negative or vice versa.

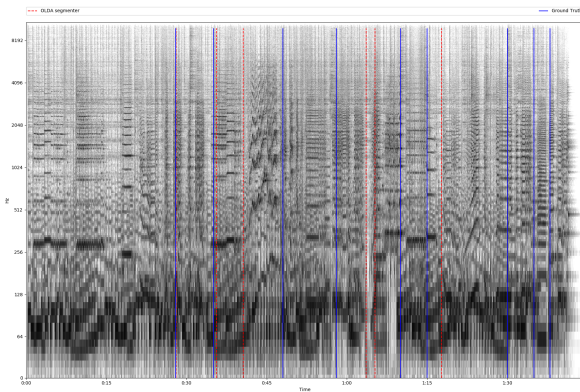


Figure 2: Comparing the ground truth (blue) and the OLDA algorithm (red) based on MFCC features.

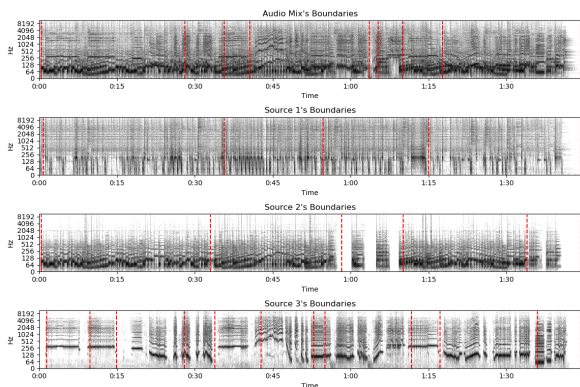


Figure 3: Segmentation boundaries on the audio mix and the audio sources, using the OLDA algorithm based on MFCC features.

## 5. DISCUSSION

The objective of this study was not to claim a methodology able to improve the current state of the art in segmentation of audio tracks, but rather to offer a perspective foregrounding the interaction of the musical voices, their contribution, and how these negotiate structure in real-time during freely improvised pieces. Neither is the objective truth used to be considered as a target for the optimization of the algorithm. Instead, it is used as an initial pruning and approximation, to choose a segmenter for the exploration described in Section 4.2. The i-events, the transitions, and the ‘sharing’ quality of the sections outlined by the method are offering an opportunity for a re-evaluation of the interpretative and cognitive process occurring when trying to infer structural dynamics in a freely improvised piece. In this sense, the method contributes suggestions, hints and viewpoints. Thus, it can be considered under the same dialogical paradigm as the music that it analyzes. For the user, this might be akin to having a conversation with another musicologist or practitioner, who would present her opinion about how the musical parts interact with one another, and how the piece emerges from such dynamics.

Inspecting the boundaries in the dynamic mode (see Figure 4), and ignoring the first clear-cut transition (which is

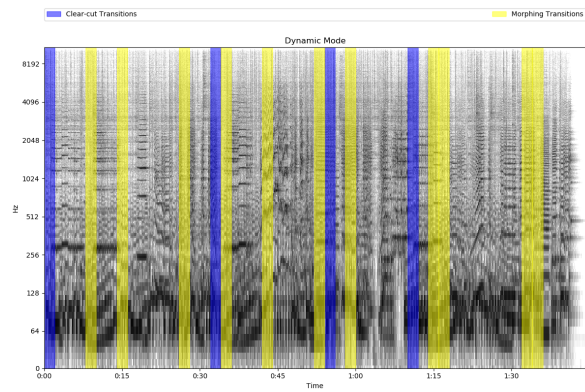


Figure 4: Dynamic Mode: clear-cut (blue) and morphing (yellow) transitions shown over the audio mix’s spectrogram.

clear-cut	32, 54, 70
morphing	8, 14, 26, 34, 42, 52, 58, 74, 76, 92, 94

Table 2: Transitions types and their activation times, in seconds

not very informative, since it is the point where the instruments start playing, at the beginning of the recording) one can consider the activation times given in Table 2.

The method so far operates with an arbitrary time kernel (or stride), whereby the total length of the piece is divided into equally spaced windows of such length. While this is heuristically determined and can be changed to one’s liking, it nevertheless produces a lower accuracy as for the boundaries’ activation times. Rather than considering this as an handicap, it is useful to be reminded that the objective of the study is to identify patterns of interaction rather than accurate segmentation boundaries. Furthermore, this allows to factor in reaction times for the musicians who, while making decisions in real-time, might change their musical behavior in the order of a few seconds. Combining the information gathered from the dynamic mode analysis, it is thus possible to infer more pronounced regions of musical interaction between the following times in seconds: 32-34, 52-58, 70-76, and 92-94. These are obtained by combining activation times that are sufficiently close together (in both the clear-cut and the morphing class). ‘Sufficiently’ is taken to mean as up to twice the time kernel, on either side.

Interestingly, these sections exhibit a lower feature similarity (see Figure 6), which might suggest that when cues are acknowledged and responded to this corresponds to a higher interaction. Conversely, in the regions in between, the individual players might adopt more consistent, static and function-oriented musical behaviors. This finding seems in accordance with the framework proposed by Pelz-Sherman, as well as with the notions of *transitions* and *relational functions/composites* in Nunn’s work [47]. To test the validity of this approach it would be appropriate to conduct further studies accounting for the opinion of



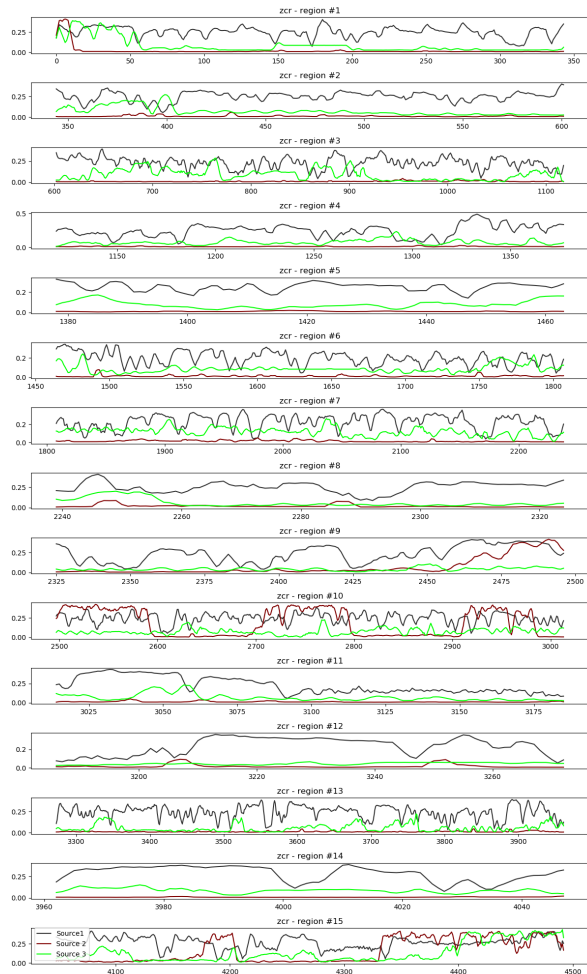


Figure 5: Inter-regions zero-crossing rate dynamics.

a wider pool of practitioners and using more recordings, since one piece cannot offer general insights. These endeavors are left for future work.

## 6. CONCLUSION

To analyze freely improvised music, it is paramount to consider the interaction between players and how their choices help shape the piece in real-time, without a predetermined plan or specific musical goals other than those arising as contingencies of the creative process. In keeping with the distributed and dialogical nature of this musical expression, this paper explores a relational stance and implements an essential method for visualizing such interactions. While mainstream musical expressions, styles and genres are more conducive to elicit well defined segments and regions (by virtue of intrinsic structural assumptions. i.e.: chorus, verse, etc.), freely improvised music requires bespoke treatment and a focus shift from surface to functional level. The current study represent but a small step in this direction, albeit partial and with a reduced scope of analysis (limited to two basic modes of interaction). Future work would benefit from implementing more discrimination within these levels, as formalized in [8, 46, 47], as

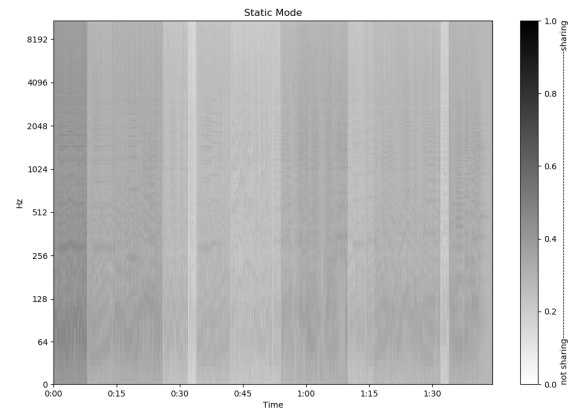


Figure 6: Static Mode: inter-regions feature space sharing.

well as from recording a dataset of multi-track freely improvised pieces. To this end, the author plans to record several duets comprising a wide range of instrumentation, with active practitioners pooled from the freely improvised music scene in UK and Japan.

## 7. REFERENCES

- [1] G. Fauconnier and M. Turner, “Conceptual integration networks,” *Cognitive Science*, vol. 22, no. 2, pp. 133 – 187, 1998.
- [2] F. Lerdahl and R. Jackendoff, *A generative theory of tonal music*. Cambridge, MA: MIT Press, 1983.
- [3] E. Jost, *Free jazz*. Universal Edition Graz, 1974.
- [4] J. Schwartz, *Free Jazz: A Research and Information Guide*. Routledge, 2018.
- [5] D. Bailey, *Improvisation: Its Nature And Practice In Music*. Da Capo Press, 1993.
- [6] E. Prévost, *No Sound is Innocent: AMM and the Practice of Self-invention, Meta-musical Narratives, Essays*. Copula, 1995.
- [7] C. Canonne and N. B. Garnier, “Cognition and Segmentation In Collective Free Improvisation: An Exploratory Study,” in *International Conference on Music Perception and Cognition*, Thessaloniki, Greece, 2012.
- [8] M. Pelz-Sherman, “Suggested Applications of Musical Improvisation Analysis to Sonification,” in *International Workshop on Interactive Sonification*, York, UK, 2007.
- [9] G. E. Lewis, “Too many notes: Computers, complexity and culture in voyager,” *Leonardo Music Journal*, vol. 10, pp. 33–39, 2000.
- [10] A. Ozerov, E. Vincent, and F. Bimbot, “A general flexible framework for the handling of prior information in

- audio source separation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 4, pp. 1118–1133, May 2012.
- [11] D. Fitzgerald, “Harmonic/percussive separation using median filtering,” in *International Conference on Digital Audio Effects (DAFx)*, 2010.
- [12] R. Bittner, J. Salamon, M. Tierney, M. Mauch, J. Bello, and et al., *MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research*, 10 2014.
- [13] S. Mimilakis, E. Cano, J. Abeber, and G. Schuller, “New sonorities for jazz recordings: Separation and mixing using deep neural networks,” in *International Conference on Music Information Retrieval*, 09 2016.
- [14] E. Perez-Gonzalez and J. Reiss, “Automatic equalization of multichannel audio using cross-adaptive methods,” in *Audio Engineering Society Convention 127*, Oct 2009.
- [15] M. D. Plumbley, S. A. Abdallah, J. P. Bello, M. E. Davies, G. Monti, and M. B. Sandler, “Automatic music transcription and audio source separation,” *Cybernetics and Systems*, vol. 33, pp. 603–627, 2002.
- [16] S. Ewert, B. Pardo, M. Mueller, and M. D. Plumbley, “Score-informed source separation for musical audio recordings: An overview,” *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 116–124, May 2014.
- [17] E. M. Grais, D. Ward, and M. D. Plumbley, “Raw multi-channel audio source separation using multi-resolution convolutional auto-encoders,” *CoRR*, vol. abs/1803.00702, 2018.
- [18] A. Honingh and R. Bod, “Convexity and the well-formedness of musical objects,” *Journal of New Music Research*, vol. 34, no. 3, pp. 293–303, 2005.
- [19] A. Forte, *The Structure of Atonal Music*. Yale University Press, 1973.
- [20] C. Roads and P. Wieneke, “Grammars as representations for music,” *Computer Music Journal*, vol. 3, no. 1, pp. 48–55, 1979.
- [21] E. Neeman, “Free Improvisation as a Performance Technique: Group Creativity and Interpreting Graphic Scores,” Ph.D. dissertation, The Julliard School, New York, 5 2014.
- [22] V. Goudard, “John, the semi-conductor : a tool for comprovisation,” *CoRR*, vol. abs/1811.06858, 2018.
- [23] R. Dudas, “Comprovisation: The Various Facets of Composed Improvisation within Interactive Performance Systems,” *Leonardo Music Journal*, vol. 20, pp. 29–31, 2010.
- [24] R. T. Dean, F. Bailes, and J. Drummond, “Generative structures in improvisation: Computational segmentation of keyboard performances,” *Journal of New Music Research*, vol. 43, no. 2, pp. 224–236, 2014.
- [25] J. Pressing, “The micro- and macrostructural design of improvised music,” *Music Perception: An Interdisciplinary Journal*, vol. 5, no. 2, pp. 133–172, 1987.
- [26] C. Canonne, “Focal Points in Collective Free Improvisation,” *Perspectives of New Music*, vol. 2, 2013.
- [27] J. Nattiez, “Fondements d’une semiologie de la musique,” *Journal of Aesthetics and Art Criticism*, vol. 35, no. 2, pp. 239–242, 1976.
- [28] S. W. Smoliar, “Music programs: An approach to music theory through computational linguistics,” *Journal of Music Theory*, vol. 20, no. 1, pp. 105–131, 1976.
- [29] J. A. Moorer, “Music and computer composition,” *Commun. ACM*, vol. 15, no. 2, pp. 104–113, Feb. 1972.
- [30] T. Winograd, “Linguistics and the computer analysis of tonal harmony,” *Journal of Music Theory*, vol. 12, no. 1, pp. 2–49, 1968.
- [31] A. M. Perlman and D. Greenblatt, “Noam Chomsky Meets Miles Davis: Some Observations on Jazz Improvisation and Language Structure,” in *The sign in music and literature*, W. Steiner, Ed. Austin, TX: University of Texas Press, 1981.
- [32] B. L. J. Smith, I. Schankler, and E. Chew, “Listening as a creative act: Meaningful differences in structural annotations of improvised performances,” *Music Theory Online*, vol. 20, no. 3, 2014.
- [33] E. D. Scheirer, “Music-Listening Systems,” Ph.D. dissertation, Massachusetts Institute of Technology, Boston, MA, 6 2000.
- [34] D.-J. Povel and E. Jansen, “Perceptual mechanisms in music processing,” *Music Perception: An Interdisciplinary Journal*, vol. 19, no. 2, pp. 169–197, 2001.
- [35] J. K. Wright and A. S. Bregman, “Auditory stream segregation and the control of dissonance in polyphonic music,” *Contemporary Music Review*, vol. 2, no. 1, pp. 63–92, 1987.
- [36] A. S. Bregman and J. Campbell, “Primary auditory stream segregation and perception of order in rapid sequences of tones,” *Journal of Experimental Psychology*, vol. 89, no. 2, pp. 244–249, 1971.
- [37] E. Cambouropoulos, “The musical surface: Challenging basic assumptions,” *Musicae Scientiae*, vol. 14, no. 2-suppl, pp. 131–147, 2010.
- [38] M. Silverstein, “Metapragmatic discourse and metapragmatic function,” in *Reflexive Language: Reported Speech and Metapragmatics*, J. A. Lucy, Ed. Cambridge University Press, 1993, pp. 33–58.
- [39] R. K. Sawyer, *Group creativity: Music, theater, collaboration*. Mahwah, NJ, US: Lawrence Erlbaum Associates Publishers, 2003.

- [40] I. T. Monson, *Saying Something: Jazz Improvisation and Interaction*. Chicago: University of Chicago Press, 1996, winner of the Sonneck Society's 1998 Irving Lowens Prize for best monograph in American music published in 1996.
- [41] R. K. Sawyer, "Improvisation and the creative process: Dewey, collingwood, and the aesthetics of spontaneity," *The Journal of Aesthetics and Art Criticism*, vol. 58, no. 2, pp. 149–161, 2000.
- [42] V. Iyer, "Exploding the narrative in jazz improvisation," in *Uptown Conversation: The New Jazz Studies*, R. O'meally, B. Edwards, and F. Griffin, Eds. New York: Columbia University Press, 2004.
- [43] F. Okiji, "Storytelling in jazz work as retrospective collaboration," *Journal of the Society for American Music*, vol. 11, no. 1, p. 7092, 2017.
- [44] G. B. Wilson and R. A. R. MacDonald, "The sign of silence: Negotiating musical identities in an improvising ensemble," *Psychology of Music*, vol. 40, no. 5, pp. 558–573, 2012.
- [45] G. Michaelsen, "Analyzing musical Interaction in Jazz improvisations of the 1960s," Ph.D. dissertation, Indiana University, Bloomington, IN, 2013.
- [46] M. Pelz-Sherman, "A Framework for the Analysis of Performer Interactions in Western Improvised Contemporary Art Music," Ph.D. dissertation, University of California, San Diego, CA, 1998.
- [47] T. E. Nunn, "Wisdom of the Impulse: On the Nature of Musical Free Improvisation," 1998.
- [48] O. Nieto and J. Bello, "Systematic exploration of computational music structure research," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 8 2016.
- [49] C. i Wang and G. J. Mysore, "Structural segmentation with the variable markov oracle and boundary adjustment," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 291–295, 2016.
- [50] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532)*, vol. 1, July 2000, pp. 452–455 vol.1.
- [51] O. Nieto and T. Jehan, "Convex non-negative matrix factorization for automatic music structure identification," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 236–240.
- [52] B. McFee and D. P. W. Ellis, "Learning to segment songs with ordinal linear discriminant analysis," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 5197–5201.
- [53] —, "Analyzing song structure with spectral clustering," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

# Mechanical Entanglement: A Collaborative Haptic-Music Performance

Alexandros Kontogeorgakopoulos

Cardiff School of Art and Design  
Cardiff Metropolitan University

akontogeorgakopoulos@cardiffmet.ac.uk georgios.sioros@imv.uio.no

George Siorros

RITMO, Department of Musicology  
University of Oslo

Odysseas Klissouras

oneContinuousLab

odysseas@onecontinuouslab.net

## ABSTRACT

Mechanical Entanglement is a musical composition for three performers. Three force feedback devices each containing two haptic faders are mutually coupled using virtual linear springs and dampers. During the composition, the performers feel each others' gestures and collaboratively process the music material. The interaction's physical modelling parameters are modified during the different sections of the composition. An algorithm which process three stereo channels, is stretching in and out-of-sync three copies of the same music. The performers are controlling the stretching algorithm and an amplitude modulation effect, both applied to recognisable classical and contemporary music recordings. Each of them is substantially modifying the length and the dynamics of the music and is simultaneously affecting subtly or abruptly the gestural behaviour of the other performers. At fixed points during the composition, the music becomes gradually in sync and the performers realign their gestures. This phasing game between gestures and sound, creates tension and emphasises the physicality of the performance.

## 1. INTRODUCTION

The computer music research community has been exploring the use of haptics and force feedback within a musical context since the first explorations in the late seventies at ACROE [1]. Numerous force feedback interfaces for musical purposes have been developed since then [2–12].

The last decade has similarly presented a growing interest in musical composition with the use of force feedback technology. Compositions such as *Running Backwards Uphill* by Hayes, *Engraving Hammering Casting* by Berdahl and Kontogeorgakopoulos, *Hélios* by Cadoz, *Quartet for Strings* by Beck, *Of Grating Impermanence* by Pfalz amongst others have explored the potential of haptics in purely musical or audiovisual artistic context [13–15]. In all of these compositions for solo musicians or small orchestras such as the Laptop Orchestra of Louisiana, there was no intercoupling at the gestural level; the hands of the musicians were mechanically coupled with their musical instruments but not between them. The current pa-

per explores this novel concept by presenting a composition based on a developed haptic digital musical instrument where the gestures of the performers are co-influenced mechanically during the performance.

Mechanical Entanglement is an electroacoustic composition and a research project on a collaborative haptic musical system. It was composed for a small musical ensemble of three performers (trio) interacting with three mutually coupled force feedback devices. The performers process in real time the same sonic material while they were interacting mechanically between them through a virtual viscoelastic network. This novel type of collaborative performance offers a new type of music co-creation based on haptic telepresence.

The paper is organised in three sections. The first section presents the technical aspects of the musical system developed. The second section offers an insight on the compositional and the performative elements of the project. The final section presents a discussion on the project holistically, both from a functional and an aesthetic point of view.

## 2. SYSTEM DESCRIPTION

A 3D model of the system's structure used in the project is illustrated in figure 1. As can be seen from this higher level description, the system consists of two main blocks with different functions. The haptic component of the system executes all the haptic signal processing operations and generates the haptic responses while the sound component executes all the audio signal processing operations and generates the audio output. The following subsection presents in more details those two components.

### 2.1 Haptic Signal Processing

The haptic device used in the research project and composition is the FireFader [11]. This device consists of open-source hardware and open-source software elements and is optimised for introducing musicians to haptics. It offers a single-degree-of-freedom motorised potentiometer fader at a low price and can be combined with the haptic signal processing framework (HSP) where the users can quickly design and develop their own haptic, audio and visual responses and hence create complete multimodal environments and compositions [16]. HSP runs on well-known computer music languages such as Max and Pure Data which was a considerable advantage for the current project since other sound processing algorithms which interact with the haptics were developed in those languages

Copyright: © 2019 Alexandros Kontogeorgakopoulos et al.  
This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



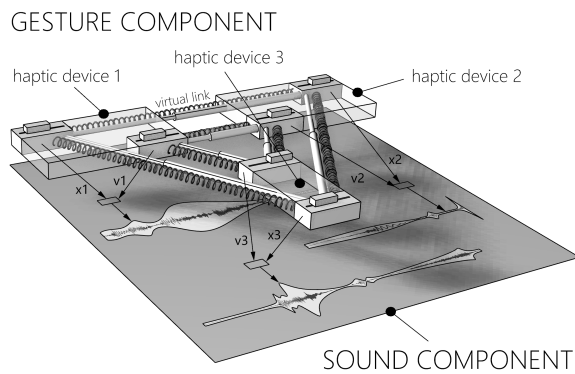


Figure 1. 3D model of the System's Structure.

too. Alternative commercial general-purpose cost-effective force feedback devices such as the NovInt Falcon from NovInt Technologies or the Geomagic Touch (formerly Phantom Omni) from 3DSYSTEMS were not considered since they do not have an appropriate form and workspace aligned with the concept of the project.

Three FireFaders with two motorised faders each are connected through USB to a computer that performs mainly the haptic calculations and runs the haptic models designed by the authors on the Max programming environment. The audio signals are generated on a different computer, connected to the first one through the open sound control protocol (OSC). Therefore data captured in real time from the haptic faders on the first computer are transmitted through an Ethernet twisted pair link into a second computer in order to control the audio playback and processing. More details regarding the audio processing part of the setup is given in the following section. Finally, each haptic device has two bright LED lights offering visual feedback of the force applied to each motor. This feature added an interesting angle to the performance which is further explored in section 3.

The physical model designed and developed for the project is based on the lumped element modelling paradigm and more precisely on the Cordis-Anima system [17]. A simple mass-interaction network, connecting linearly the three haptic devices between them was implemented on the Haptic Signal Processing framework using the Max programming environment. Each faders physical knob behaves like an ideal material element in the virtual mechanical network. Those ideal masses are linked between them mechanically using linear springs and dampers. The spring constants and the damping coefficients are modified dynamically during the length of the composition as each section corresponds to a different set of parameters. Figure 2 presents the block diagram of the overall signal processing system developed in this project.

Therefore the three haptic devices are part of the same oscillatory system. The performers are controlling the same sound processing algorithms with their gestures while they are "internally" interacting between them via the virtual

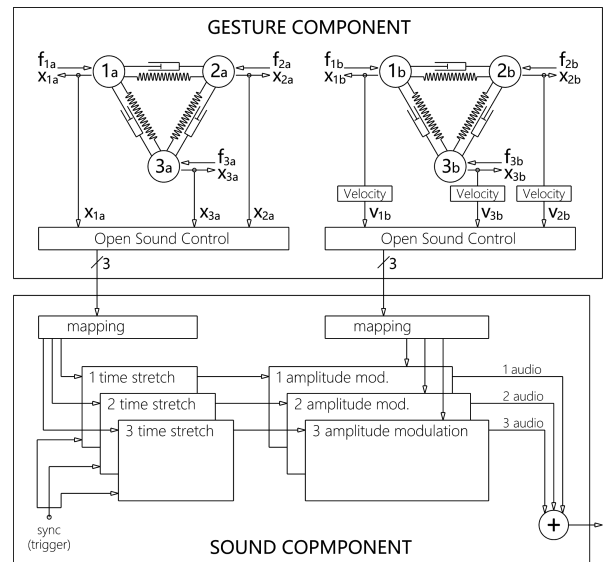


Figure 2. Block Diagram.

links in the virtual network. Their hands are in continuous mechanical interaction which creates a mysterious reciprocal influence. The whole process is coherent energetically because of the nature of the force feedback devices and their coupling with the physical models. This is what Cadoz calls *ergotic interaction* [18]. The OROBORO controller futures some similar aspects where they introduced a haptic mirror in which the movement of one performers sensed hand is used to induce movement of the partners actuated hand and vice versa [19].

## 2.2 Music Signal Processing

All sound is produced through the Ableton Live digital audio workstation with the embedded Max For Live environment (Max4Live). Therefore, the Live software forms the space in which the composition is created in a traditional linear timeline and at the same time hosts all the necessary software that enable the faders to control the performance through the OSC messages. This software was created as two Max4Live software devices, that is, special Max/MSP patches that can be loaded as plugins in the Ableton Live environment.

Our composition was created as a Live Set, that is, the type of document that you create and work within Ableton Live. It comprises three identical but independent of each other tracks corresponding to the three pairs of haptic faders. Each track hosts the two Max4Live devices serially connected that we developed for the purposes of this composition. The first device plays back a predefined audio file at a variable playback speed without affecting its pitch, while the second controls the amplitude of its output. Each pair of faders controls a single track. The position of the first of the two faders in the pair controls the playback speed, so that it varies between normal to the extreme 0.1 x the original speed, which resembles a freeze effect. The velocity of the second fader controls the amplitude, so that when the fader does not move no sound is coming out of



the tracks output. A linear mapping between velocity and amplitude was used with an adjustable scale factor.

All three tracks play the same audio file which is pre-loaded into memory using a shared buffer Max object. Playback starts at the beginning of the file and in synchrony between the three tracks. However, as the playback speed varies according to the position of each fader, a phase difference between the tracks accumulates. However, each playback device is equipped with a sync button, which sends its playback position to the other playback devices (through send-receive Max objects), effectively forcing playback in all tracks to align again. Before the tracks jump to the new sync position, they smoothly fade out leaving only a single track sounding for a short duration of about 2 sec, before all tracks start playing back together in synchrony. The amplitude dropping not only helps avoiding undesirable cuts in the sound but it also emphasizes the synchronization.

Besides controlling the performance, the Live Set was used to structure the composition into sections. To this end, a separate track was used to send OSC messages to the haptic faders system through a dedicated Max4Live device. In this way, we could choose dynamically or according to the timeline the parameters of the physical models corresponding to each section of the composition. Since our composition comprises several sections that need different audio files, several memory buffers were used to load the audio files of each section at the beginning of the performance, ensuring a smooth transition between the sections. The playback devices were simply switching between memory buffers at the beginning of each section.

### 3. COMPOSITION

#### 3.1 Ideas and Concept

The music project is based on the concept of stretching: physically-stretching a simulated material while simultaneously time-stretching a pre-recorded music material. It aspires an integration of auditory, haptic and even visual cues with use of dimmed LED lights as we will see in the following section. Figure 3 illustrates metaphorically what was happening at a mechanical level during the performance.

Each performer is allowed to move at his own speed by the time-stretching algorithm through the given material, a process which Michael Nyman calls *people process* [20].

The tensions created by this process, form the "sculptural" elements of the musical composition. The action-reaction pair of forces from all the performers, acting together on the same musical parameters is creating points of equilibrium, sonically and gesturally. The principal compositional process employed is *phasing* [21]. The time-stretching algorithm simultaneously processes three stereo audio channels, stretching in an out-of-sync three copies of the same music recording.

The simultaneous playback of the same material at different speeds creates tension both harmonically and rhythmically [22]. This tension is resolved when the playback between the tracks is forced to realign at certain moments in

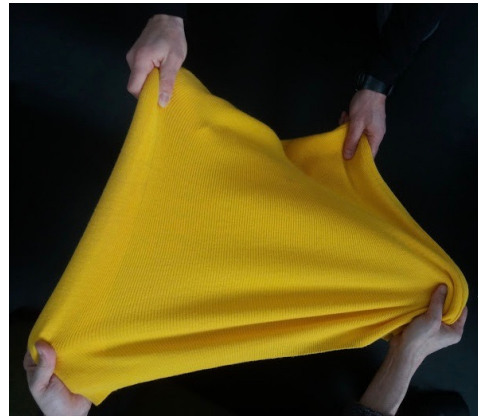


Figure 3. A metaphor: hands stretching a physical material.

the composition (through the sync button on the playback Max4Live devices described in section 2.2). This resolution does not last long as the playback drifts anew between the tracks. Nevertheless, these moments of ephemeral synchronisation function as structural anchors in the composition.

Rhythmically, layering the same rhythmic pattern at three different speeds presents the listener with three different possible metrical frameworks, each at a different tempo and phase. Some listeners might focus on the stream that has a moderate tempo (in the range 80-130 bpm) as Parnutt has shown, [23], or on the stream that is closer to the original speed and therefore more recognisable. As the speed and tempo of each stream varies, the attention focus of the listener constantly shifts between them.

At the same time, each tracks rhythm is the result of the interaction between the amplitude modulation and the rhythm resulting from the playing back at a variable tempo. However, even though each tracks audio is manipulated independently, both the tempo and amplitude modulation are the result of a single physical model realised in the network of haptic faders. The interaction between the performers is what drives the variance between the tracks. After all, until the performers interact through the faders no sound is produced as the tracks move forward through the audio files silently and in synchrony. Any tension arising from layering different versions of the same material reflects the tension and forces between the faders arising from the physical model system connecting the performers.

#### 3.2 Performance in the Gallery

Mechanical Entanglement was performed in M.A.D.E. gallery in Cardiff in June 2016. It was not a typical concert situation, which created a captivating experience both for the performers and the audience. The composers of the piece, also the performers in that occasion, tried to create a serene environment which matched the quiet nature of the composition. They sat on cushions on the floor in close proximity to each other, each holding a haptic device on their hand while the audience sat or stood around them. The room was dark enough in order to emphasise further



Figure 4. The performance in the gallery space.

the force feedback activity with the use of responsive LED lights as mentioned in section 2.1 and depicted in figure 4. This created a dramatic atmosphere, where the hands of the performers were cast with light in the moments of the performance with high physical tension. A camera operator was also filming the process and projecting it on a gallery wall.

The music material may vary each time the piece is performed. The following list is the selection of music tracks that were used completely or partially in the performance in M.A.D.E. in the order of appearance. The total duration of the concert was approximately 30min.

- *Pavane pour une Infante Defunte* by Maurice Ravel
- *Superman* by Laurie Anderson
- *Cello Suite no1* by Johann Sebastian Bach
- *Drumming pt. II* by Steve Reich
- *Prelude and Fugue No 1 C* by J. S. Bach
- *Symphony No 6, 5th movement* by L. van Beethoven
- *Blue Moon* by Elvis Presley
- *Bolero* by Maurice Ravel

Each section of the composition corresponded to a different music track and had its own physical modelling parameters. Therefore it allowed different type of gestural interaction, from very fast ones to smooth and precise ones where the nuances amongst the performers were felt more intensively. A few excerpts from the performance can be heard on the following link <sup>1</sup>, as recorded from a stereo microphone positioned in the middle of the room of the

<sup>1</sup> <https://onecontinuouslab.net/Projects/#MechanicalEntanglement>

gallery space. The music tracks were chosen according to how they responded to the time-stretching algorithm and their overall texture. The authors preferred to include well-known tonal musical compositions from different genres with clear and preferably repetitive musical structure and moderate dynamic range.

It is interesting to mention that during the performance, the modelling coefficients often took values impossible to occur in nature such as negative damping between the interaction of the performers, thus creating very unfamiliar interaction sensations. Moreover, instabilities that occurred due to the long feedback control delays made the gestural and sonic dialogue very difficult and quite often unpredictable. This is a difficult problem in haptic interaction and is related to the latency between the hardware and the software components of the device. The total latency all the way around the control loop with audio running in Max ranges between 7ms and 15ms due to jitter as measured and reported on the firmware of the FireFader device. In the current occasion the composers decided to use creatively these instabilities and make them part of the compositional and performance discourse.

#### 4. DISCUSSION

The performers constantly shaped and explored a "viscoelastic" environment of gestures and sound. In the physical-tactile level they were always feeling the flow of interactions between them and had to find ways of anticipating the unpredictability of their instrument behaviour. The fingertips functioned simultaneously to express the performers own musical intention and experience the intentions of others. As such, the act of performing was indispensably connected with the act of tactile-listening, forming an enhanced tactile environment, where every performing force is applied upon forces produced by the other performers. By participating in this mass-interaction network the performers could perceive themselves as active counter weights on an oscillatory system, which had no fixed point at all. This intercoupling at the gestural level provided a haptic telepresence where each performer preserved a unique "view point" or *tact-point* as authors called them, of the performance space.

The auditory experience was common for every performer; by aligning the performers to a common task gave the shift for an intense collaborative group experience to happen. The auditory level acted as a catalyst for the abstract and mental aspects of the composition to permeate into the physical level. The performers were challenged to focus on the flow dynamics of the group's interaction environment, instead of solely mastering a deterministic musical instrument.

This approach is aligned with Chadabe's taxonomy of electronic musical instruments as a continuum between deterministic and indeterministic function [24]. The current project fluctuates continuously between those states, since the shared control with the other performers (and not with algorithms as in Chadabe's case) often gives the impression that the instrument in itself generates unpredictable information to which the performers have to react.

A minor problem of the design of the haptic devices was that the auditory experience was interrupted by sounds made by the haptic device itself, when the fader was reaching its end points during abrupt oscillation moments. This problem can be solved partially by designing a more robust enclosure for the haptic device, a direction which the team has started to explore after the performance <sup>2</sup>.

From the point of view of the audience, the LED lights provided a pleasant visual feedback of the interactions and gestural activities but often proved inadequate to offer a coherent understandable connection between the composition and the haptic system. However,, they reveal the systems idiosyncratic nature at moments of apparent inactivity but with strong counter forces at the fingertips of the performers.

It would be interesting to recreate the same conditions in the future, with haptic digital audio effects as described in [25]. In this scenario, the physical audio effect models implemented likewise with the mass-interaction physical modelling paradigm would provide force feedback to the performer without any disruption of the energetic loop between the performers, the haptic device and the physical models. The audio processing and haptic processing algorithm belong to the same physical model. Therefore the performers would be fully immersed in a mechanical network that would equally produce the haptic responses and the audio output without any "artificial" mapping strategy. Finally network music performances and further research on the topic of haptic telepresence within the music context are planned in the future.

### Acknowledgments

The authors would like to thank the Fab-Cre8 multidisciplinary research and enterprise group at Cardiff Metropolitan University for their financial support and MADE gallery in Cardiff for hosting the performance. This work was partially supported by the Research Council of Norway through its Centres of Excellence scheme, project number 262762.

## 5. REFERENCES

- [1] C. Cadoz, A. Luciani, J.-L. Florens, and N. Castagne, "ACROE-ICA: Artistic Creation and Computer Interactive Multisensory Simulation Force Feedback Gesture Transducers," in *Proceedings of the 2003 conference on New Interfaces for Musical Expression*. Montreal, Canada: McGill University, 2003, pp. 235–246.
- [2] C. Cadoz, A. Luciani, and J.-L. Florens, "Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms: The Cordis System," *Computer Music Journal*, vol. 8, no. 3, pp. 60–73, 1984.
- [3] B. Gillespie, "The Touchback Keyboard," in *Proceedings of the International Computer Music Conference*, San Jose, CA, 1992, pp. 77–80.
- [4] C. Nichols, "The vbow: A Virtual Violin Bow Controller for Mapping Gesture to Synthesis with Haptic Feedback," *Organised Sound*, vol. 7, no. 02, pp. 215–220, 2002.
- [5] B. Verplank, M. Gurevich, and M. Mathews, "The Plank: Designing a Simple Haptic Controller," in *Proceedings of the 2002 conference on New Interfaces for Musical Expression*. Dublin, Ireland: Media Lab Europe, May 2002, pp. 1–4.
- [6] J. Carlile and B. Hartmann, "OROBORO: A Collaborative Controller with Interpersonal Haptic Feedback," in *Proceedings of the 2005 conference on New interfaces for musical expression*. Vancouver, Canada: University of British Columbia, 2005, pp. 250–251.
- [7] T. H. Andersen, R. Huber, A. Kretz, and M. Fjeld, "Feel the Beat: Direct Manipulation of Sound during Playback," in *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, Adelaide, Australia, Jan. 2006, pp. 123–126.
- [8] R. Oboe, "A Multi-Instrument, Force-Feedback Keyboard," *Computer Music Journal*, vol. 30, no. 3, pp. 38–52, 2006.
- [9] J. Rodriguez, A. Shahrokni, M. Schrittenloher, and M. Fjeld, "One-Dimensional Force Feedback Slider: Digital Platform," in *IEEE VR 2007 Workshop on Mixed Reality User Interfaces: Specification, Authoring, Adaptation*, 2007, pp. 47–51.
- [10] R. Gabriel, J. Sandsjo, A. Shahrokni, and M. Fjeld, "BounceSlider: Actuated Sliders for Music Performance and Composition," in *Proceedings of the Second International Conference on Tangible and Embedded Interaction*, Bonn, Germany, Feb. 2008, pp. 127–130.
- [11] E. Berdahl and A. Kontogeorgakopoulos, "The fire-fader: Simple, open-source, and reconfigurable haptic force feedback for musicians," *Computer Music Journal*, vol. 37, no. 1, pp. 23–34, 2013.
- [12] J. Bak, W. Verplank, and D. Gauthier, "Motors, Music and Motion," in *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, 2015, pp. 367–374.
- [13] E. Berdahl and A. Kontogeorgakopoulos, "EngravingHammeringCasting: Exploring the Sonic-Ergotic Medium for Live Musical Performance," in *Proceedings of the International Computer Music Conference*, Ljubljana, Slovenia, Sep. 2012, pp. 387–390.
- [14] L. Hayes, "Performing Articulation and Expression Through a Haptic Interface," in *Proceedings of the International Computer Music Conference*, 2012.
- [15] J. Leonard, N. Castagne, C. Cadoz, and A. Luciani, "The MSCI Platform: A Framework for the Design and Simulation of Multisensory Virtual Musical Instruments," in *Musical Haptics*, S. Papetti and C. Saitis, Eds. Springer, 2018, pp. 151–169.

<sup>2</sup> <https://onecontinuouslab.net/Projects/#HapticFaderEnclosure>

- [16] E. Berdahl, A. Kontogeorgakopoulos, and D. Overholt, "HSP v2: Haptic Signal Processing with Extensions for Physical Modeling," in *Proceedings of the Haptic Audio Interaction Design Conference*, Copenhagen, Denmark, Sep. 2010, pp. 61–62.
- [17] C. Cadoz, A. Luciani, and J.-L. Florens, "CORDIS-ANIMA: A Modeling and Simulation System for Sound and Image SynthesisThe General Formalism," *Computer Music Journal*, vol. 17, no. 1, pp. 19–29, 1993.
- [18] C. Cadoz and M. Wanderley, "Gesture-Music," in *MM Wanderley and M. Battier (Eds.), Trends in gestural control of music.* Ircam, 2000.
- [19] K. McDonald, D. Kouttron, C. Bahn, J. Braasch, and P. Oliveros, "The Vibrobyte: A Haptic Interface for Co-located Performance," in *Proceedings of the 2009 conference on New interfaces for Musical Expression.* Pittsburgh, Pennsylvania, USA: Carnegie Mellon School of Music, 2009.
- [20] M. Nyman, *Experimental music: Cage and beyond.* Cambridge University Press, 1999, vol. 9.
- [21] S. Reich, *Writings on Music, 1965-2000.* Oxford University Press, 2002.
- [22] E. F. Clarke, "Levels of Structure in the Organization of Musical Time," *Contemporary Music Review*, vol. 2, no. 1, pp. 211–238, 1987.
- [23] R. Parncutt, "A Perceptual Model of Pulse Salience and Metrical Accent in Musical Rhythms," *Music Perception: An Interdisciplinary Journal*, vol. 11, no. 4, pp. 409–464, 1994.
- [24] J. Chadabe, "The Limitations of Mapping as a Structural Descriptive in Electronic Instruments," in *Proceedings of the 2002 conference on New Interfaces for Musical Expression.* Dublin, Ireland: Media Lab Europe, 2002, pp. 1–5.
- [25] A. Kontogeorgakopoulos and G. Kouroupetroglou, "Low Cost Force-Feedback Haptic Interaction with Haptic Digital Audio Effects," in *Lecture Notes in Artificial Intelligence: Gesture and Sign Language in Human-Computer Interaction and Embodied Communication*, E. Efthimiou, G. Kouroupetroglou, and S.-E. Fotinea, Eds., vol. 7206. Springer Verlag, 2012, pp. 48–57.



# State Dependency - Audiovisual interaction through brain states

**Patrick Neff**

Centre for Neuromodulation,  
Department of Psychiatry and Psychotherapy,  
University of Regensburg,  
Germany  
patrick.neff@ukr.de

**Jan Schacher**

Institute for Computer Music  
and Sound Technology (ICST),  
Zurich University of Arts,  
Switzerland  
jan.schacher@zhdk.ch

**Daniel Bisig**

Institute for Computer Music  
and Sound Technology (ICST),  
Zurich University of Arts,  
Switzerland  
daniel.bisig@zhdk.ch

## ABSTRACT

Artistic installations using brain-computer interfaces (BCI) to interact with media in general, and sound in specific, have become increasingly numerous in the last years. Brain or mental states are commonly used to drive musical score or sound generation as well as visuals. Closed loop setups can emerge here which are comparable to the propositions of neurofeedback (NFB).

The aim of our audiovisual installation *State Dependency*, driven by brain states and motor imagery, was to enable the participant to engage in unbound exploration of movement through sound and space unmediated by one's corpo-reality. With the aid of an adaptive feedback loop, perception is taken to the edge.

We deployed a BCI to collect motor imagery, visual and cognitive neural activity to calculate approximate entropy (a second order measure of neural signal activity) which was in turn used to interact with the surround *Immersive Lab* installation. The use of entropy measures on motor imagery and various sensory modalities generates a highly accessible, reactive and immediate experience transcending common limitations of the BCI technology.

*State dependency* goes beyond common practice of abstract routing between mental or brain with external audiovisual states. It provides new territory of unrestrained kinaesthetic and polymodal exploration in an immersive audiovisual environment.

## 1. INTRODUCTION

Interaction with sound and image without a direct physical action is possible with the aid of BCIs. With this technology, 'imagined' activity, so called *motor imagery* can be directly linked to sound and image and influence the perception of motion in space. Placed in an immerse audiovisual environment, such a technically mediated perceptual cycle can provide a clear experience of an intentional, but suppressed movement and its reality to the mind. This shows that human perception of agency and the emergence of reality is constructed by the brain and completed in a

multimodal fashion, even if certain sensory elements are missing.

Media technology allows the simulation of a physical surrounding through a few well placed and carefully balanced sound and image elements. This functional and mostly modality-specific simulacrum of a naturally occurring phenomenon may induce the brain to enter into resonance with and to reinforce the perception of the most credible scenario through completion [1].

*State Dependency* aims at realising a NFB scenario through a research experiment with artistic techniques and scientific methods. In the feedback between the (preparatory) imagination of movement and the perception of motion, the blending of modalities becomes possible, in a way that goes beyond established functional combinations of sensory inputs.

The adaptive feedback configuration proposed is based on the premise that perception depends on both inner and outer states and is dependent on reactive behaviour. In this experimental approach, the reactive loop is modulated to extend the limits of normal perception and (re-)action cycles.

Concretely, we want to combine movement control via motor imagery [2] and audio-visual media in a spatial setup. Driven by entropy measures, these two layers can coalesce into a coherent multi-modal immersive experience that is rooted in kinaesthetic and visual perception.

The inner sense of position and movement in space of the participant gets projected or linked onto a physically present, enveloping audio-visual flow of particles. The resulting amalgamated experience points toward the concept of 'telekinesis'. Thus, the installation can be experienced as a participant-centric telekinetic steering of the energetic particle flow. Alternatively, it can be felt as a physical trans-location inside a stream, where the sense of position and centrality gets lost, even if momentarily.

The merging of the modes of position, agency, and movement sensing lead to a new experience and awareness of interaction with – projected – reality. This pre-reflective, sub-personal access to agency and a sense of position and orientation proposes a mode of experience that explores poly-modal kinaesthetically-driven awareness, hitherto an uncharted territory.

Copyright: © 2019 Patrick Neff et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 2. BACKGROUND

BCIs, mostly in the form of commercial ready-to-wear electroencephalography (EEG) devices, have become widely-used in (digital) art and music [3].<sup>1</sup> In these installations, brain-derived EEG signals are measured on the head, amplified and sent to a recording or streaming device (e.g., computer or mobile device). The raw signal or processed derivatives are then used to control visual or acoustic features of the installation.

Given the plethora of recent and current works using BCI in the art and music context, we here focus on relevant works using real-time brain signals mapped to audio-visual outputs in a performative setting. The first report of BCI art is from 1965 where Lucier performed ‘Music for Solo Performer’ which can be considered the first brain-driven musical piece [4]. Pioneering visuals, Nina Sobell in her work ‘Brain Wave Drawings’ [5] overlaid a real-time video portrait of two persons with the EEG raw signal reflective of putatively synced brain states of the portrayed persons.

Following advances in technology, especially regarding portability of systems, BCI art became more common in the early years of the third millennium. In a recent work by Dmitry Morozov named ‘eeg\_deer’, the brain activity of the user was used to generate music and visuals in real-time with no further actions by the user [6]. In Novellos performance titled “Fragmentation” [7], on the other hand, the user performed with dance while the BCI and the linked audio-visual streams served as an extension. Combining live video footage with brain activity of users, Ursula Damm manipulated the degree of abstraction of video streams in ‘Chromatographic Orchestra’ [8].

Further adding aspects of immersion and interaction, current works also apply virtual reality (VR) technology. In ‘The Hidden Rooms’ [9] and the ‘Errant Eye’ [10] virtual (visual but not acoustic) reality is explored driven by brain states. Lastly, ‘Conductor’ [11] combines virtual and augmented reality with a BCI in a mobile setting. The setup depends on the GPS location and produces an artificial world on the screen of a smartphone alongside sounds driven by EEG and movement data.

The stability and quality, but also the reliability and validity of these measured frequency band power values are partly questionable. This is especially true when they are used in interactive closed loop setups like many of the existing BCI art installations. This is partly explained by the equipment employed, especially low-grade consumer devices with dry electrodes, but also by the situation (movement and other artefacts) and inappropriate use. In neuroscientific literature, consensus has arisen that the signal quality but also related claims of access and control over mental states are lacking any solid evidence and theoretical background [12].

Our goal was to set up an immersive 3D environment where accessible states can be explored using BCI technology in a functioning interdependent manner.

<sup>1</sup> <https://bci-art.tumblr.com/>

## 3. BRAINSTATES

In the last section, we introduced the current state of use of BCI systems in art. Concretely, we also shortly discussed technical details and limitations of BCI signal acquisition, processing and mapping, and also looked at fundamental claims of the relation between neural activity and subjective mental states. Because we identified several shortcomings or even questionable claims, the central motivation of the work at hand was to implement a working BCI installation which builds on verified aspects of current mobile neural technology.

First, motion control via motor imagery is an established concept which is used effectively for lateral bodily movements (e.g., left motor cortex  $\mu$ -rhythm de-activation to control the right arm) [13]. Usually, with electrodes placed over positions of the motor cortex,  $\mu$ -rhythm de-activation is then measured during motor imagery and compared to idle baseline activity of the same electrode. This procedure can be both time-consuming and unreliable, as the contrast between active and passive activation patterns can be too weak or disturbed by artefacts or general signal quality. The resulting baseline or calibration period can be considered as detrimental for any spontaneous setup and immersion into an art installation. In recent times, attempts have been made to optimise the measurement of this motor activity by applying machine learning (ML) on the features of motor imagery with a respective optimisation of the former differential approach [14].

Yet, the latter approach is in need of individualised baseline measurements and respective investment of computing power and time to train the ML algorithms. Therefore, we planned to use an entropy-based, second order information-theoretical analysis of ongoing neural activity to enable us to overcome the baselining as well as calibration phase and immediately start to derive motor imagery data from the activity at the electrode. This becomes possible because the  $\mu$ -rhythm is highly responsive and, more importantly, regular so that the entropy algorithm identifies the rhythm as highly ordered and thus non-entropic. The detection properties of the entropy algorithm come in handy for movement control and have already been similarly implemented in state of the art BCI applications [15]. Details about our implementation can be found in the next sections (4).

Second, while motor imagery is able to control movement or directional parameters of the audiovisual flow, we were interested in aesthetic mappings of the sensory input domain, namely the visual system. We considered the implementation of this sensory neural input as feasible for a mapping beyond movement control, as the visual cortex is a large, accessible site with high activity. With the mapping of entropy measures derived from visual cortex activity, we envision a closed-loop NFB between visual effects projected and the response of the visual system. Naturally, this feedback loop is not directly controllable by the participant but rather relies on the unsupervised reactive neural activity (reflective of visual processing) in the individual’s visual cortex.

Lastly, in an attempt to additionally experiment with cog-

nitive states, we incorporated an electrode on the forehead measuring frontal, potentially cognition- and memory-related, activity. Again, by using entropy instead of frequency band power, we aimed at covering a more global state of cognition, which contrasts ordered putatively goal-oriented active cognition with more entropic idle states of thinking. We furthermore opted to use this remaining electrode of the current BCI system to both test for mental control option and to be able to access a ‘control’ electrode signal (e.g., muscle noise artefacts from frowning). This signal could then be used to test the entropy algorithm’s vulnerability to noise or general responsiveness.

The implementation of entropy measures for motor imagery and NFB features in *State Dependency* can be considered a novel territory of artistic and scientific exploration.

## 4. DEPENDENCIES

### 4.1 Technical Implementation

The hardware and software setup of *State Dependency* provides the technical means for establishing a feedback loop between a participant’s audio-visual perception and mental activity and the generation and spatialisation of audio-visual media (see figure 1).

The setup consists of a wearable EEG BCI, a computer for acquiring and analysing EEG data, and a second computer for generating real-time audio-visual content. The BCI is based on the *OpenBCI Ganglion*<sup>2</sup> board to which multiple wet electrodes are connected. Two of these electrodes provide reference and ground signals and are attached to the Lobule and Scapha of the right ear (see figure 2). The other four electrodes are embedded into an elastic headband and mounted at the back, forehead, and crest of the head. These sensors measure neural activity in the visual cortex, pre-frontal area, and left and right primary motor cortex. The interface, which is worn around the neck, sends the sensory signals via Bluetooth to a computer. The setup of the BCI with the current wet electrodes takes about 10 minutes before the user can properly interact with the system. This computer runs the *OpenBCI GUI* application which conditions the sensor data. Conditioning involves removal of signal values around 50 Hz by a notch filter to eliminate interference from AC current. For the motor imagery data, frequencies below 8 Hz and above 14 Hz were filtered out with a band pass filter to better access the  $\mu$  rhythm. The visual and prefrontal data was filtered with a wider band pass filter between 5 Hz and 50 Hz to capture a wider range of frequencies typically produced by the areas underlying the respective electrodes. With this filtering strategy measuring artefacts, that would appear due to facial and bodily movement (i.e. below 5 Hz and above 50 Hz) as well as unwanted neural activity in different frequency bands, are suppressed. The conditioned sensor signals are subsequently sent via the Open Sound Control [OSC] protocol to an additional application that conducts feature extraction.

<sup>2</sup> <http://docs.openbci.com/Hardware/07-Ganglion>

This application has been custom designed in the *OpenFrameworks* C++ programming environment and implements the *Approximate Entropy* [AE] method according to the mathematical description provided by Sabeti et al. [16] (see figure 3). *Approximate Entropy* is a statistic measure for the predictability of new values in a time series based on the history of previous values [17]. For our purpose of controlling through neural activity the generation of video and audio in real-time, this entropy measure proved to be more responsive than other entropy measures such as *Shannon Entropy* or *Spectral Entropy*. Optimal performance of the AE algorithm was achieved by setting the size of the sample window to 512, the embedding dimension to 2, and the threshold to 1.0. Latency and responsivity are issues to be considered in this optimisation process, which was mostly performed by manual approximation resulting in the final values above. Taking into account all elements of the feedback system from measuring the activity on the scalp to the audiovisual presentation in the IL, the resulting latency is estimated at about 350-500 ms. This includes time for data transfer, parsing, processing, and feedback (e.g. visual delay of the beamers of about 100 ms). Considering aspects of volition and cortical initialisation the latency value is slightly increased. Overall, we estimate the latency not to be larger than 1 second for all the processes described above. Certainly, the system will be further optimised to decrease latency and increase responsivity for better immersive experiences.

### 4.2 Experimental Setting

An essential part of the experimental setting for this project is its application in an immerse media space called the *Immersive Lab*<sup>3</sup>. The installation was developed by two of the authors as a platform for artistic creation and research. It combines panoramic video and surround audio with touch-based interaction across its entire projection surface [18, 19]. The installation’s video projection screens are arranged into a circular setup with a diameter of about 4 meters. It is furthermore equipped with 16 loudspeakers, arranged in two rings of eight loudspeaker placed above and below the screens and complemented by two subwoofers.

The capability of the installation to fully immerse participants plays a crucial role in establishing a feedback loop between EEG controlled media generation and movement perception (see figure 4). The EEG entropy measures control the creation of audio-visual media presented in a surround format to the participant in real-time.

### 4.3 Media

#### 4.3.1 Audio

The auditory side of the media aims at providing a flow of sound elements that envelop the listener. The continuous flow of sound particles reinforces an immersed, almost hypnotic state. Changes in brain states modulate sound elements, in their balance, their spatial distribution, and their timbre. In our setup, the sound parameters are

<sup>3</sup> <http://immersivelab.zhdk.ch>

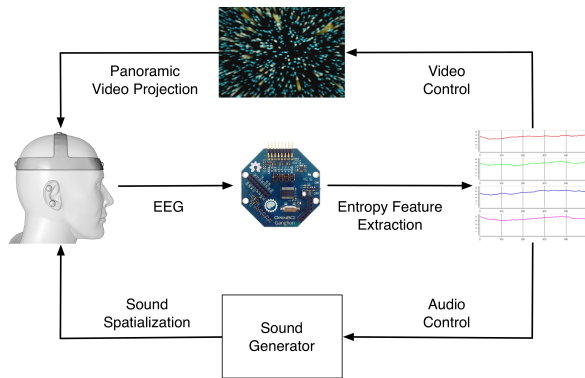


Figure 1. Schematic depiction of NFB loop. The sequence of EEG data acquisition and processing is shown in the middle section. From left to right: electrodes mounted to the participant's head, four channel EEG interface, approximate entropy-based analysis. The top and bottom sections depict the visual and acoustic feedback channels, respectively. From right to left: EEG-based control of media generation, panoramic media projection

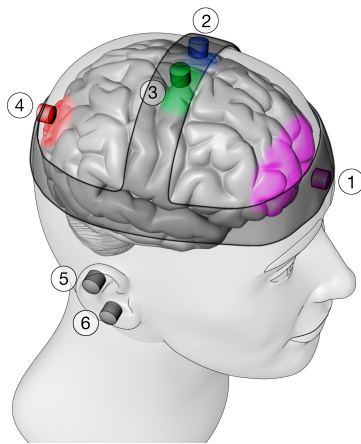


Figure 2. Schematic depiction of electrode placement. Six wet electrodes are mounted to the participant's head. Four electrodes are attached to an elastic headband and located above the following brain regions: 1. prefrontal area, 2. left primary motor cortex, 3. right primary motor cortex, 4. visual cortex. Two additional electrodes serve as reference and ground and are mounted with tape to the ear's 5. lobule and 6. scapha, respectively.

tightly linked to the flowing particles' behaviour and reinforce their physicality: A constant flow of sonic particles is streaming past the listener, always synchronised with the visual elements. Two continuously fluctuating instrumental voices provide a second, musical layer.

The sound particles are generated via granular synthesis of filtered noise. A bandpass filter is applied to the noise sources and serves two functions: The filter produces an individual pitch for each particle. The frequency of the filter is swept downwards as each particle moves past the listener, creating a Doppler effect. The particles are spatialised around the listener using Ambisonic process-

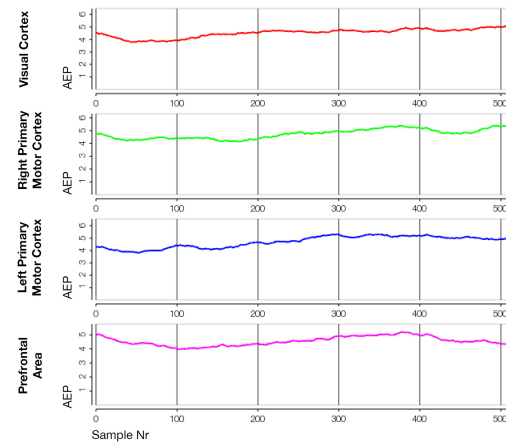


Figure 3. Approximate entropy feature analysis. Shown are entropy values as time series for all four electrodes.

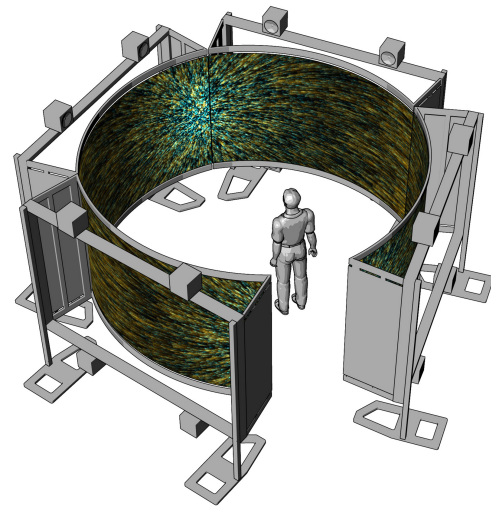


Figure 4. Schema of the installation setup. Two rings of speakers produce spatialised sound in azimuth and elevation. A panoramic screen presents an enveloping image.

ing [20], which provides a clear localisation of the sounds in the actual space of the installation.

The musical sounds originate from double bass recordings of continuous bowing. One consists of bow hair noise near the bridge of the instrument and the other stems from extended techniques providing multiple harmonics. The sound's pitches get altered through changes in playback speed, to provide a slightly alien harmonic timbre. The balance between these two sound layers is controlled by the difference in AE between left and right motor cortex.

The density of the sonic field duplicates the effect of the visual, where paying attention to a single event becomes difficult, i.e., the auditory stream segregation fails [21]. This increases the sensation of immersion and contributes to the perception of egocentric self-motion.



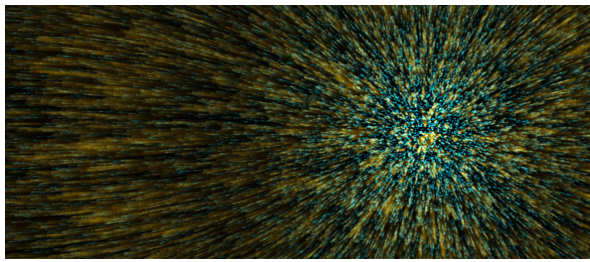


Figure 5. Screen capture of visual feedback. The image shows a section of the graphical rendering of a particle field which is four times as wide as shown here.

#### 4.3.2 Video

The imagery evokes the sensation of being in an infinite space, densely populated by a stream of visual particles in the viewer. The particles of identical appearance exhibit a uniform global movement (see figure 5). This creates a perception that lies at the limit of a participant's capability for visual stream segregation. Depending on the density and visual representation of the particles' motions, the participant's attention is either drawn towards the individual visual elements or instead evokes an illusion of egocentric imbalance and translocation. By relating EEG measurements of neural activity to changes in the particles' motion and appearance, the balance between egocentric and allocentric movement perception is altered.

The visual flow is created by drawing 500'000 particles as textured billboards whose size, colour, and transparency vary with distance. The motion of the particles is visually accentuated by superimposing the most recent image with previously drawn images. This causes a motion blur effect whose strength depends on the amount of transparency applied to the previous images.

The particles move through an euclidean space that is delimited by a spherical boundary whose origin coincides with the center of the installation. When particles exit this boundary, their positions are wrapped around to slightly randomised positions on the opposing side of the boundary surface.

In order to create a panoramic projection for the circular screen, the image goes through two stages of optical and geometrical correction. A cube map captures the space and gets transformed into a panoramic image by an equirectangular projection shader. The shader's output is mapped onto four *Bezier* surfaces which compensate for the curvature of the projection screens.

Several parameters of the particle visualisation are controlled by EEG signals acquired from the participant (see figure 6). The difference in the AE levels of the EEG signals acquired from the right and left primary motor cortex controls the direction and amount of lateral movement of the particles. The AE level of the primary visual cortex controls the amount of motion blur in the visual image. The AE level of the prefrontal area controls the distance threshold of the particle's transparency and thereby alters the visual density of the particle system.

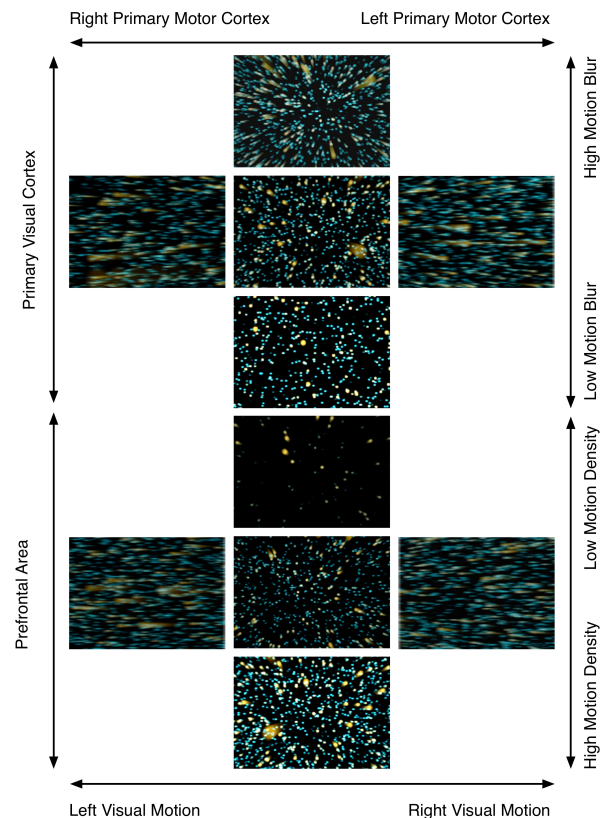


Figure 6. Overview of mapping between EEG features and visual control parameters. Approximate entropy analysis values of all four electrodes are mapped to visual control parameters as follows. The neural activity difference between the right and left primary visual cortex controls the horizontal direction of the particle's movements. Neural activity in the primary visual cortex is proportional to the amount of motion blur in the visual output. Neural activity in the prefrontal area is inversely proportional to the distance-based transparency increase of the particles.

## 5. EXHIBITION

Some visual impressions of a NFB interaction situation within the *Immersive Lab* are provided in figures 7 and 8. A video recording of this interaction situation is available online.<sup>4</sup> Notably, the installation has merely been tested by a single user which renders the following statements and claims anecdotal in nature at this point in time. It is though envisioned to continue with the installation and make it accessible to general public by working on all technical aspects of the BCI, data processing, mapping and audiovisual feedback.

During the setup, calibration and first experimental phases of *State Dependency*, the chosen NFB setting could demonstrate its potential in creating an immersive, volitional movement control and (mental) state dependent experience. The specific combination of state of the art technology with a consistent, intuitively-accessible installation enables immediate entrance to polymodal movement, drift

<sup>4</sup> [http://immersivelab.zhdk.ch/?page\\_id=4485](http://immersivelab.zhdk.ch/?page_id=4485)

and flow through audiovisual space without intermediate physical borders. The motor imagery setup is responsive and the interaction with it can be quickly integrated in one's own mental movement perception, while the separate visual NFB loop allows the participant to de-focus and even lose track of the granularity of the audiovisual output. The visual blurriness nicely emphasises this visual and possibly related attentional de-focus. Regarding cognitive control, we did not expect much intentional control options through mental imagery or cognition, which also became obvious when actually interacting with the setup. This comes as no surprise given the complexity of human cognition and related neural correlates - yet may be an interesting avenue for further iterations of this installation and BCIs in general.

An important novel insight generated by this installation was the the observed mechanism of a kinaesthetic sense strongly influenced by the visual motion. This mechanism can enforce the perceived movement immensely and therefore acts as a cross-modal 'interaction between visual, auditory and kinetic modalities. It can thus be very difficult to neutralise or change directions, when exposed to the visual motion with ever increasing speed (i.e., to move in the opposite direction than the visual and kinaesthetic modalities are implying). This challenge in the actual movement control has to be learned to keep a balance and flexibility in the movement interaction with the setup.

All in all, this novel immersive closed loop setup has shown to be successful in enabling the participant to freely interact with an audiovisual surrounding blended with his motor imagery, sensory and mental states. With time, the participant can reach a state of dependency with the linked system, which extends his self-concept and agency beyond the usual physical and mental boundaries.



Figure 7. Exhibition situation in the Immersive Lab. A participant controls audio-visual media through a NFB loop

## 6. DISCUSSION

### 6.1 Perception

In the immersive, interactive situation of this installation, the participant is able to explore kinaesthetic and proprioceptive modalities in their pure form: No intermediate motor system is required to alter movement and in consequence its perception. In practice, the participant there-

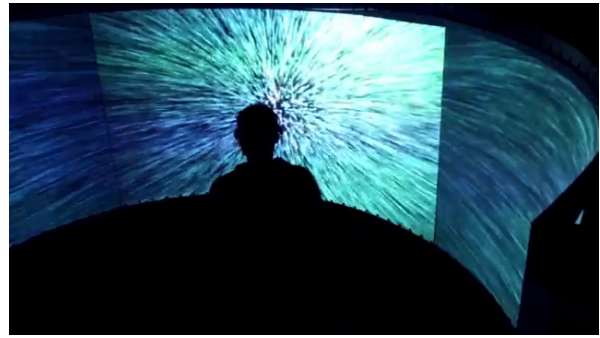


Figure 8. Overview of the exhibition situation in the Immersive Lab. A participant controls audio-visual media through a NFB loop

fore merely imagines (rotary) movements on a basic, pre-conscious or intuitive level, while at the other end of the feedback loop the media-counterpart of this imagination is manifested. It is surprising, how intuitively this motor imagery can be learned and applied. However, this impression emerged during the testing of the installation by a single user. The setup has to be validated with other users to further back these experiences. Certainly, the responsiveness of the system contributes critically to this effect and thus also further increases immersion.<sup>5</sup>

A fundamental question that becomes apparent in this setting is about the contradiction inherent to presenting simulated content in a physical media installation. From an 'enactive' point of view [22], the participant is put into a state of suspension, in order to eliminate the disparity between physical space and simulated perception. From a phenomenological point of view [23], however, this setting makes sense: the presented stimuli reach a level of consistency, and are convincing enough for the perception to fill in the missing pieces. The suspension of disbelief on the one hand, and the physical embeddedness in the immersive setup on the other, allows the participant to reach the edge of coherent perception and experience the breakdown and error-correction of movement control and spatial movement perception.

The goal is to take perception to the edge, with the aid of an adaptive feedback loop. This loop involves the participant's intention, through the motor imagery of motion, and affects the perception of stability in space, which is both a kinaesthetic and a proprioceptive perception. There is a contradiction between the visual and auditory flow and the physiological stability in place in the centre of the installation. The inner perceptual achievement of suspension allows for a sense of effortless movement through space. The real-time nature of the adaptive loop contributes to the sensation of agency. Perceiving that one is 'the author of change in the environment' is usually physiologically linked through the coherence between the efferent and afferent sensory streams tied to an action [24]. In this configuration, providing agency is achieved by circumventing direct action and directly taking the cortical arousal state to

<sup>5</sup> For a short video documentation see:  
<https://tube.switch.ch/videos/e815b5e7>

modify the media stimuli presented to vision and audition.

Critical for the success of this effect is a responsive mapping and relaying of the felt inner states to the external audiovisual stream. Not only should the setup be reactive in the split-second temporal resolution, but also cover a meaningful range, which can be thoroughly explored. Only then does the setup become an extension or dissolvent of the participant's corporeal boundaries.

## 6.2 Implementation

On the hardware and setup level, the applying of the EEG BCI system remains critical for a reliable and valid data flow from the brain to the subsequent devices. Dry electrodes have not been able to convince any EEG expert in respect to signal quality. Impedances below 10 k $\Omega$  at the electrode are not considered feasible for any EEG application, especially not for real time signals. In the case of dry electrodes, impedances can be as high as >100 k $\Omega$  and therefore the measured signal is highly unreliable [12]. Notably, signal quality or vulnerability to external noise sources increases proportionally with the impedance. Wet electrodes, on the other hand, require a more tedious and time-consuming mounting, which certainly limits their use in artistic contexts, especially in public installations or performances.

Even with an implementation of second order computations on the signal like entropy measures, which are less susceptible to bad signals, stable and reliable signals are key for a proper setup. This is particularly true when it comes to scales and ranges of mapped parameters, which have to be carefully optimised to allow for a satisfying immersion, as it is aimed at in the case of our installation. Of special note, we can not generalise our settings to a broader audience at this point because our installation has been set up and calibrated for a single individual. Beyond electrode montage difficulties, impedances and individuality issues, further artefacts can be present in the signal. Most prominently, and inevitable in an unconstrained media installation, are movement artefacts. There are also issues with the wireless data transmission streaming (mostly Bluetooth) given the spatial extent and interference present in many installation venues.

Taken together, these technical aspects still pose some limitations on the endeavour of using BCIs as live data for art installations.

## 7. OUTLOOK

It is conceivable that changing the currently fixed relationship between mental states and media content to an evolving one could increase immersiveness and alleviate the habituation and saturation process taking place in any artificial scenario. On the other hand, a quick succession of changes might prove to be counterproductive. After all, achieving through an adaptive loop an inner state of agency and telekinesis relies on a carefully calibrated configuration of techniques and connections. Another interesting modification of the setup could involve the generation of

more naturalistic audiovisual content as part of the adaptive feedback loop.

On the technical side, the basic setup would profit from a faster and easier montage of the electrode headband. Additionally, exact positioning, especially in the case of the motor cortex electrodes, and individual differences are critical for optimised data quality. The responsiveness of the EEG entropy feature extraction could also be improved to better handle the participant's willingness to change movement directionality. Such optimisation could not only be performed on the parameters inherent to the AE calculation but also be implemented as an additional layer of signal analysis (e.g., rate of change of AE difference).

Given the relative stability of the BCI headset construction and measures of artefact and noise control in the EEG signal processing pipeline, it is conceivable that the participant can also make use of the touch features of the *Immersive Lab*. This could add an interesting optional layer of possibly further explorable amalgamation of the fed back states and modalities.

Thinking ahead, the chosen approach possesses great application potential in the context of VR. Being able to control through imagined rather than physical locomotion one's own movement through a virtual environment constitutes a fundamentally different approach to currently existing navigation and locomotion interfaces for VR. Traditional locomotion interfaces place the user of a VR system into a constraining physical exertion device such as an omni-directional treadmill or sliding surface within which he or she is barely able to execute natural locomotion behaviours. BCI-based interfaces don't require such a setup since they abolish the need for actual physical locomotion but instead tap into the brain's innate capability to supplant actual motor activity with imagined motor activity. The preliminary results that have been described in this paper indicate that a BCI-based interface in combination with highly correlated and spatialised audiovisual feedback can successfully supplement physical and vestibular self-motion cues and thereby offer a level of movement control and awareness that is likely more natural and intuitive than is the case for traditional mechanical locomotion interfaces.

Finally, given the current state of the project and its evaluation, the installation would profit from experiences from different users. Ideally, technical functionality should be tested with different age groups and their respective technological backgrounds to investigate individual differences and generalisability of the system. A survey on both the control aspects but also on the subjective, immersive experience is envisioned to better evaluate the installation. Lastly, given the slow visual movements and absence of flickering or flashes, health issues (e.g. epilepsy or nausea) are very unlikely. General reported side effects of NFB are unsystematic and only reported in very rare cases with non-critical symptoms. On the other hand, the situation in the installation and the relying on NFB principles, may additionally qualify *State Dependency* as a relaxing or meditative experience.



## Acknowledgments

The Immersive Lab is a long term project at the Zurich University of the Arts. The current project cycle is funded by the Swiss National Science Foundation, AGORA Grant Nr. RAGP1 171656. Patrick Neff holds an Early-PostDoc Grant from the Swiss National Science Foundation (P2ZHP1\_174967).

## 8. REFERENCES

- [1] B. Nanay, "The importance of amodal completion in everyday perception," *i-Perception*, vol. 9, no. 4, p. 2041669518788887, 2018.
- [2] J. Schacher, "Motor imagery in perception and performance of sound and music," in *Oxford Handbook on Sound and Imagination*, M. Grimshaw, M. Walther-Hansen, and M. Knakkegaard, Eds. Oxford, UK: Oxford University Press, 2019.
- [3] E. R. Miranda and J. Castet, *Guide to brain-computer music interfacing*. Springer, 2014.
- [4] V. Straebel and W. Thoben, "Alvin Lucier's "Music for Solo Performer": Experimental music beyond sonification," *Organised Sound*, vol. 19, no. 1, pp. 17–29, Apr. 2014. [Online]. Available: <http://www.journals.cambridge.org/abstract.S135577181300037X>
- [5] N. Sobell, "Brainwave Drawings," 1973. [Online]. Available: <http://colophon.com/ninasobell/parkbenchdocs/portfolio/3/frame.html>
- [6] D. Morozov, "eeg\_deer," 2014. [Online]. Available: <http://vtol.cc/filter/works/eegdeer/>
- [7] A. Novello, "Fragmentation." [Online]. Available: <http://jestern.com/>
- [8] U. Damm, "Chromatographic Ballads." [Online]. Available: <http://ursuladamm.de/nco-neural-chromatographic-orchestra-2012>
- [9] M. F. Bojanowski, "Hidden Rooms." [Online]. Available: <http://artistsinlabs.ch/portfolio/marie-france-bojanowski>
- [10] A. Dunning, P. Woodrow, and M. Hollenberg, "The Einstein's Brain Project ." [Online]. Available: <http://people.ucalgary.ca/~einbrain/new/main.html>
- [11] Aramique, "Conductar." [Online]. Available: <http://aramique.com/conductar-moogfest>
- [12] A. Wexler and R. Thibault, "Mind-reading or misleading? assessing direct-to-consumer electroencephalography (eeg) devices marketed for wellness and their ethical and regulatory implications," *Journal of Cognitive Enhancement*, pp. 1–7, 2018.
- [13] H.-J. Hwang, K. Kwon, and C.-H. Im, "Neurofeedback-based motor imagery training for brain-computer interface (bci)," *Journal of neuroscience methods*, vol. 179, no. 1, pp. 150–156, 2009.
- [14] M. Krauledat, K. Grzeska, M. Sagebaum, B. Blankertz, C. Vidaurre, K.-R. Müller, and M. Schröder, "Playing pinball with non-invasive bci," in *Advances in neural information processing systems*, 2009, pp. 1641–1648.
- [15] W.-Y. Hsu, "Assembling a multi-feature eeg classifier for left-right motor imagery data using wavelet-based fuzzy approximate entropy for improved accuracy," *International journal of neural systems*, vol. 25, no. 08, p. 1550037, 2015.
- [16] M. Sabeti, S. Katebi, and R. Boostani, "Entropy and complexity measures for eeg signal classification of schizophrenic and control participants," *Artificial intelligence in medicine*, vol. 47, no. 3, pp. 263–274, 2009.
- [17] J. Bruhn, H. Röpcke, and A. Hoeft, "Approximate entropy as an electroencephalographic measure of anesthetic drug effect during desflurane anesthesia," *Anesthesiology: The Journal of the American Society of Anesthesiologists*, vol. 92, no. 3, pp. 715–726, 2000.
- [18] B. Milosevic and J. Schacher, "The Immersive Lab: An interactive audio-visual space for artistic exploration," in *Proceedings of Korean Electro-Acoustic Music Society's 2015 Annual Conference (KEAMSAC2015)*, Seoul, Korea, 2.–3. October 2015, pp. 35–40.
- [19] J. Schacher and D. Bisig, "Haunting Space, Social Interaction in a Large-Scale Media Environment," in *Human-Computer Interaction - INTERACT 2017: 16th IFIP TC 13 International Conference, Mumbai, India, September 25–29, 2017, Proceedings, Part I*, ser. Lecture Notes in Computer Science (LNCS), R. Bernhaupt, G. Dalvi, A. Joshi, D. K. Balkrishan, J. O'Neill, and M. Winckler, Eds. Cham, Switzerland: Springer International Publishing, 2017, pp. 242–262.
- [20] M. Neukom and J. C. Schacher, "Ambisonic Equivalent Panning," in *Proceedings of the International Computer Music Conference ICMC, Copenhagen, Denmark*, 2008.
- [21] A. S. Bregman, *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, MA: MIT Press, 1994.
- [22] E. Myin and J. Degenaar, "Enactive Vision," in *The Routledge Handbook of Embodied Cognition*. London, UK, New York, NY: Routledge, 2014, pp. 90–98.
- [23] S. Gallagher, "Phenomenology and Embodied Cognition," in *The Routledge Handbook of Embodied Cognition*, L. Shapiro, Ed. London, UK, New York, NY: Routledge, 2014, pp. 9–18.
- [24] A. Berthoz and J.-L. Petit, *The Physiology and Phenomenology of Action*. Oxford, UK: Oxford University Press, 2008.

# Perceptual Evaluation of Modal Synthesis for Impact-Based Sounds

**Adrián Barahona**

Department of Computer Science,  
University of York  
ajbr501@york.ac.uk

**Sandra Pauletto**

Department of Media Technology  
and Interaction Design,  
KTH Royal Institute of Technology  
pauletto@kth.se

## ABSTRACT

The use of real-time sound synthesis for sound effects can improve the sound design of interactive experiences such as video games. However, synthesized sound effects can be often perceived as synthetic, which hampers their adoption. This paper aims to determine whether or not sounds synthesized using filter-based modal synthesis are perceptually comparable to sounds directly recorded. Sounds from 4 different materials that showed clear modes were recorded and synthesized using filter-based modal synthesis. Modes are the individual sinusoidal frequencies at which objects vibrate when excited. A listening test was conducted where participants were asked to identify, in isolation, whether a sample was recorded or synthesized. Results show that recorded and synthesized samples are indistinguishable from each other. The study outcome proves that, for the analysed materials, filter-based modal synthesis is a suitable technique to synthesize hit sound in real-time without perceptual compromises.

## 1. INTRODUCTION

Nowadays, most video games, films and pieces of media are sound designed using pre-recorded samples. Pre-recorded samples are obtained from direct audio recordings or layered sound effects and stored in audio files. However, pre-recorded samples have several limitations in interactive environments such as video games. As actions in games can be performed several times, if there is only one sample per action, the sound will be repeated, which can lead to listener fatigue and loss of authenticity [1]. To solve this, several samples can be assigned and shuffled played when a player performs an action, which the consequential increment in studio and implementation time, asset management and memory footprint problems.

Hit or impact-based sounds are the acoustic consequence of physical collisions. Changes in an object material or size will produce changes to the resulting impact sound. For games or interactive applications with hundreds or thousands of interactable assets, such as open world games or VR experiences, this will lead to an exponential growth in

the need for different sound samples to sonify any particular scenario where two or more of those assets collide.

In the context of video games and interactive applications, an alternative solution is to use real-time sound synthesis during gameplay. This approach, known as procedural audio, is defined by Farnell [2] as a “non-linear, often synthetic sound, created in real time according to a set of programmatic rules and live input”.

Farnell [2] enumerates several benefits procedural audio has over pre-recorded samples. Among them, the programmatic nature of procedural audio allows sound designers to decide or change aesthetic considerations later in the development cycle. Moreover, as procedural audio is object based, it can automate part the sound design process, especially in the implementation stage, as a single procedural audio model can contain all the possible sonic interactions a player can perform. Procedural audio also offers more variety, versatility and adaptability than pre-recorded samples. While a pre-recorded sample will always play the same way, procedural audio can change dynamically.

However, Farnell [2] also identifies perceived realism as one of the problems in procedural audio. Synthesized sound effects can often be perceived as too synthetic compared to pre-recorded samples. One of the challenges is, then, to create procedural audio models that can be indistinguishable from pre-recorded samples.

This study aims to measure whether or not it is possible to identify synthesized hit sound effects using filter-based modal synthesis. Instead of comparing pre-recorded samples and their synthesized versions side by side, this study will present them independently. The motivation of taking this approach relies in the idea that if pre-recorded hit samples and efficient real-time synthesized hit sound effects are indistinguishable from each other, the synthesized version can be used without perceptual loss of authenticity and obtaining all the benefits procedural audio presents. The null hypothesis of this study is that the synthetic sound effects are easily recognizable from the recorded ones.

### 1.1 Previous work

There is a body of research on evaluating the perception of synthesized sounds. A recent study analysed different sound synthesis techniques for different sound classes (applause, babble, bees, fire, rain, stream, waves and wind), concluding that there is not substantial difference between the reference sample and the synthesized version when an appropriate synthesis method was used, except for additive

Copyright: © 2019 Adrián Barahona et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

synthesis [3]. However, they did not focus on impact-based sounds.

Other studies evaluated the use of modal synthesis for the synthesis of weapons sounds [4]. The results showed a convincing result for weapons with resonant modes (such as axe, hammer or rapier). However, they took a rating approach for the experiment, where participants rated several versions of the sound effects (synthesized, synthesized and processed and pre-recorded) instead of evaluating them in isolation. The motivation of evaluating the sounds in isolation instead of rating several versions of them comes from avoiding any possible bias from comparing a specific sound to another. Real-time synthesized rolling sounds of glass and wood were also evaluated in other experiments [5]. Instead of identifying synthetic samples, participants rated the synthesized sounds in a 0-100 scale to evaluate the realism obtained in the rolling effect.

Tests to discriminate whether a sound is recorded or synthetic have been developed in the field of synthesis of musical instruments. Wun, Horner and Ayers [6] proposed a discrimination factor,  $d$ , which measure the quality of a synthetic tone based on how often it can be distinguished from its recorded counterpart. The discrimination factor is similar to the *Accuracy* rate used in other works dealing with binary classifiers [7]. This metric has been used to evaluate the use of synthetic piano sustain-pedal effects [8] or a synthetic clavinet model [7].

## 2. METHOD

### 2.1 Sound Synthesis

Filter-based modal synthesis is a particular use of subtractive synthesis and it is especially indicated to synthesize impact-based sounds [9]. Filter-based modal synthesis has two components: a deterministic part, the model modes, and a stochastic part, the model noise envelope.

The process consisted in analysing a pre-recorded sample to extract its modes. Modes are the individual sinusoidal frequencies to which an object vibrates, in this case, when it is impacted. In the spectrogram of a hit sound, these modes are represented by straight horizontal lines (Figure 1). The modes are used as frequency bands in a series of resonant filters which band-pass an enveloped white noise signal. These modes represent the deterministic part of the sound.

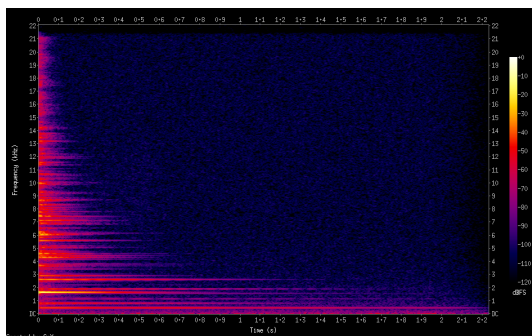


Figure 1. Recorded metal flask hit spectrogram.

The modes are then subtracted from the original sound, obtaining the noise envelope, also known as the residue. This represents the stochastic component of the signal. The residue is stored in an audio file and it is triggered with the deterministic component.

The ChucK programming language [10] was used for the modes extraction and residue generation. The extraction and residue generation code was written by Perry Cook [11].

For this study, nine sounds were recorded from materials that exhibit clear modes when excited. The choice of the materials was driven by the suitability of the synthesis method used. These materials were ceramic (2 samples: a plate and a mug), glass (3 samples: an empty bottle, a water glass and a pint glass), metal (3 samples: two lids of different sizes and a flask) and wood (1 sample: a short rod) (Figure 2). The hits were performed with a metal spoon. The recordings were made at 44.1kHz/24bit with a Zoom H6 recorder, using the built-in XY capsules. 100 modes were extracted from each material and subtracted from the original recording to generate the residue.



Figure 2. Materials used.

The modal synthesizer was also programmed in ChucK, using a modified version of a modal synthesizer created by Cook [11]. Each sound was synthesized by using an enveloped white noise signal through 100 resonant band-pass filters plus the residue. Every time a hit is triggered, the individual mode frequencies, individual mode gains, individual filter Q, residue pitch (playback speed) and balance between the deterministic and stochastic components were randomized. The level of randomization differs slightly for each material. For context, in the case of the mug, the range of randomization of the individual frequencies was of frequency  $\pm$  frequency/300, of the individual gains was of gain  $\pm$  gain/5, the individual filter Q were randomized between 800 and 1200, the residue playback rate was randomized between 0.99 and 1.01, the gain of the deterministic component was randomized between 5 and 30 and finally the gain of the stochastic component was randomized between 0.7 and 1. The aim of this randomization is to create a natural variation between hits.

A live performance of the modal synthesizer was recorded. One audio file between 3 and 6 seconds with a series of hits was recorded for each object. The original non-synthetic recordings were sliced to generate one audio file for each object comparable to the synthesized version. The synthesized versions did not have any reverberation as they

were not recorded in any physical environment. To avoid any bias in the perceptual evaluation, an impulse response of the room where the original recordings were made was recorded and mixed with the synthesized samples. All audio files were normalized to 0LU and exported in Ogg Vorbis. The choice of Ogg Vorbis instead of uncompressed WAV was determined by the technical limitations of the questionnaire platform used, Qualtrics<sup>1</sup>. However, this should not cause a drastic perceptual variation [12]. No more processing was applied to the audio files.

The recorded sounds, the ChucK code of the synthesizer used and the resulting synthesized sounds can be found in the online repository<sup>2</sup>.

## 2.2 Experimental design

The test used was inspired by the RS -or real and synthetic-listening test proposed by Gabrielli, Squartini and Vlimki [7]. Although the test was originally used with musical instruments, it can be easily applied to sound effects. The RS listening test proposes a series guidelines and this study does not follow all of them.

In this study, as opposed to the RS guidelines, the test was not carried out in a controlled listening environment. The listening test was conducted online with no information of the playback device used by the participants, although the use of headphones was suggested. This helps to replicate more closely the playing environment, where the participants are likely to use their own equipment to play the game or interactive application. The participants were asked to identify, one by one, whether the sound played was recorded or synthesized (Figure 3). Participants classified the samples without being asked to specify to which material a sample belongs to. The order of the audio samples was randomized and the participants were asked to listen to each sound just once. As suggested in the RS test guidelines, an acid test (a clearly synthesized sound) was included to acts as a control.

The participants were also asked to introduce their level of expertise in sound design, ranked from 1 (no expertise) to 5 (professional).

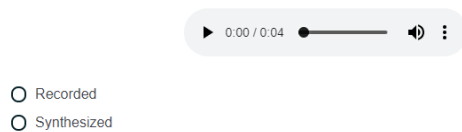


Figure 3. Online test interface.

The metrics used were the discrimination factor and the F-measure.

The discrimination factor,  $d$ , is described as [6]:

<sup>1</sup> <https://www.qualtrics.com>

<sup>2</sup> Repository containing the recorded sounds, the ChucK code of the synthesizer used and the resulting synthesized sounds: <https://github.com/adrianbarahona/SMC-Conference-2019-Perceptual-Evaluation-of-Modal-Synthesis-for-Impact-Based-Sounds>

$$d = \frac{P_{CS} - P_{FP} + 1}{2} \quad (1)$$

With  $P_{CS}$  being the percentage of correctly detected synthesized sounds and  $P_{FP}$  the false positives (recorded samples identified as synthetic). Following the RS test criteria,  $d$  values below 0.75 mean the sounds compared are considered indistinguishable from each other. Values of  $d$  around 0.5 are not different from random guessing.

As suggested in the RS guidelines, the F-measure was also evaluated. F-measure is described as [7]:

$$F - measure = \frac{(\beta^2 + 1) \times Precision \times Recall}{\beta^2 \times Precision + Recall} \quad (2)$$

With *Recall* defined as:

$$Recall = \frac{P_{CS}}{P_{CS} + P_{FN}} \quad (3)$$

And *Precision* defined as:

$$Precision = \frac{P_{CS}}{P_{CS} + P_{FP}} \quad (4)$$

Being  $P_{FN}$  the percentage of false negatives (synthetic sounds labeled as recorded) and  $\beta = 1$ . The interpretation of the F-measure values is similar to the  $d$  values.

A total of 19 participants, 12 males and 6 females (1 participant did not disclose this information) with ages between 18 and 58 took the test. A breakdown of the participants level of expertise in sound design is showed in Figure 4.

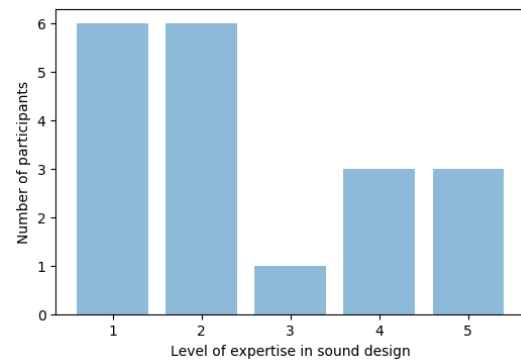


Figure 4. Participants level of expertise in sound design.

## 3. RESULTS

All participants correctly labelled the acid test as synthetic, so it was removed from the analysis of results. The purpose of this sample was to filter out participants that were random guessing or not paying attention to the test.

The mean results clearly show that recorded and synthetic hit sound effects are indistinguishable from each other. All scores are below the 0.75 threshold and closer to the random guessing mark of 0.5. There is also a low fluctuation

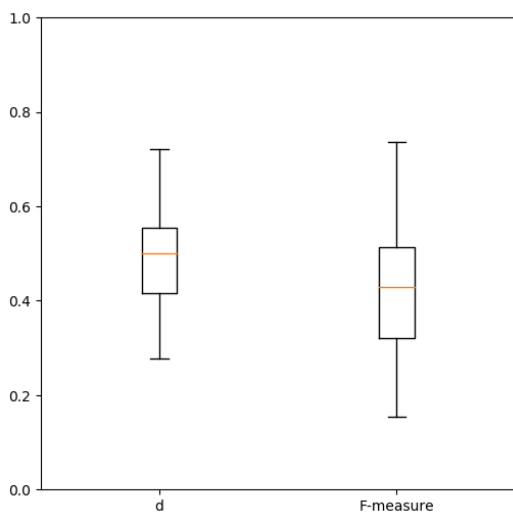


Avg $d$	$\sigma$	$\sigma^2$	Max $d$	Avg F-measure	$\sigma$	$\sigma^2$	Max F-measure
0.5	0.12	0.02	0.72	0.43	0.16	0.02	0.73

Table 1.  $d$  and F-measure values across all participants.

Samples	% Correctly labeled as recorded	Samples	% Correctly labeled as synthesized
Ceramic recording	66%	Ceramic synthesized	29%
Glass recording	54%	Glass synthesized	37%
Metal recording	60%	Metal synthesized	40%
Wood recording	68%	Wood synthesized	68%

Table 2. Different materials raw results.

Figure 5.  $d$  and F-measure values across all participants.

between the participants (Figure 5). The results are shown in the Table 1.

Some samples performed better than others. The synthetic version of the wood rod was successfully identified by 13 of the 19 participants. In the other hand, the synthetic version of the mug was only identified by 4 of 19. An overall breakdown of the different materials performance is shown in Table 2. The table shows the different objects grouped in the correspondent material.

The level of expertise in sound design was not a decisive factor to spotting synthetic sounds. Participants with an expertise in sound design ranked between 1 and 2 (out of 5) scored a  $d$  and F-measure of  $d = 0.48$  ( $\sigma = 0.10$ ) and F-measure = 0.42 ( $\sigma = 0.14$ ). Participants with the highest level of expertise in sound design, 4 and 5, scored  $d = 0.55$  ( $\sigma = 0.11$ ) and F-measure = 0.46 ( $\sigma = 0.17$ ). In fact, the participant with the highest  $d$  and F-measure (0.72 and 0.73 respectively), had no expertise in sound design.

#### 4. DISCUSSION

This paper presented the perceptual evaluation of filter-based modal synthesis hit sounds using a method inspired in the RS listening test. The aim was to measure whether or not listeners can identify synthesized hit sound effects

using filter-based modal synthesis. Results showed that, for the analysed materials, recorded and synthetic samples are indistinguishable from each other.

The different performance among the materials suggest a further study focused on what specific materials are more suitable for this particular synthesis method. Moreover, a bank of modes and their relative residue files could be created, removing the need of analysing a new audio file for each new model. The bank of modes can be created by analysing several recordings from the same material to establish a range of common modes for each material analysed.

Filter-based modal synthesis is comparable to the method used by Mengual, Moffat and Reiss to synthesize weapon sounds [4]. In their case, they took an additive approach, using spectral modelling synthesis [13]. Instead of using filtered white noise for the deterministic component, they used sinusoidal waves and the noise component is also modelled instead of triggered from the residue file. Filter-based modal synthesis offers more control over the deterministic component as the parameters of the filters, such as the Q, can be controlled in real time. However, filter-based modal synthesis offers less control over the stochastic component in this case, as it is extracted directly from the original recording and stored as an audio file.

There are some improvements to the modal synthesiser than can be implemented and evaluated. First, to ease the CPU usage, a test measuring the perceptual impact of using less modes can be performed. The synthesizer programmed for this paper uses 100 modes for each material, but it would be beneficial to draw threshold in the number of modes where models start losing authenticity. This could be also applied to implement dynamic levels of audio detail in video games. Another test could measure where that threshold is situated when the procedural models are not played in isolation but as part of a soundscape, given that frequency masking could hide their synthetic nature.

Another improvement can be the use of individual filter frequency decay time as every individual mode decay time is different for each material. This effect can be clearly appreciated in Figure 1. A test could measure the perceptual impact of this feature, comparing models with and without it. This is especially interesting for materials that performed worse using the current synthesizer, such as wood. In addition, the residual component could be also modeled



with filtered white-noise by taking the most relevant modes from the residue file.

The study results combined with the fact that the synthesizer runs in real time encourages the use of the models within a game engine. This can be done in the Unity game engine [14] by using the Chunity plugin [15]. Chunity is a package for Unity that integrates ChucK within the game engine. Tests to measure the synthesizer impact on the CPU at runtime could be done to determine whether the models are ready for production.

Controlling the synthesizer in real-time can also be a direction for a second stage of this study. Physical parameters within the game engine such as stiffness, size or geometry have been already used to control the sound of modal synthesizers in real-time [16]. This opens the possibility of expanding the use of the models to perform other actions apart from hits, such as scratching or brushing, or using haptic controllers to interact with the synthesizer.

### Acknowledgments

This work was supported by the EPSRC Centre for Doctoral Training in Intelligent Games & Game Intelligence (IGGI) [EP/L015846/1].

### 5. REFERENCES

- [1] R. Selfridge, D. Moffat, E. Avital, and J. Reiss, "Creating Real-Time Aeroacoustic Sound Effects Using Physically Informed Models," *Journal of the Audio Engineering Society*, vol. 66, no. 7/8, pp. 594–607, 2018.
- [2] A. Farnell, "An Introduction to Procedural Audio and Its Application in Computer Games," in *Audio Mostly Conference*, 2007, pp. 1–31.
- [3] D. Moffat and J. D. Reiss, "Perceptual Evaluation of Synthesized Sound Effects," *ACM Transactions on Applied Perception*, vol. 15, no. 2, pp. 1–19, 2018.
- [4] L. Mengual, D. Moffat, and J. Reiss, "Modal Synthesis of Weapon Sounds," in *Audio Engineering Society Conference: 61st International Conference: Audio for Games*, London, UK., 2016.
- [5] E. Murphy, M. Lagrange, G. Scavone, P. Depalle, and C. Guastavino, "Perceptual Evaluation of a Real-time Synthesis Technique for Rolling Sounds," in *Conference on Enactive Interfaces*, Grenoble, France, 2007.
- [6] C.-W. Wun, A. Horner, and L. Ayers, "Perceptual Wavetable Matching for Synthesis of Musical Instrument Tones," *Journal of the Audio Engineering Society*, vol. Vol.49, no. 4, pp. 250–262, 2001.
- [7] L. Gabrielli, S. Squartini, and V. Välimäki, "A Subjective Validation Method for Musical Instrument Emulation," in *AES 131st Convention*, New York, USA, 2011.
- [8] H.-M. Lehtonen, H. Penttinen, J. Rauhala, and V. Välimäki, "Analysis and Modeling of Piano Sustain-Pedal Effects," *The Journal of the Acoustical Society of America*, vol. 122, no. 3, pp. 1787–1797, 2007.
- [9] P. R. Cook, *Real Sound Synthesis for Interactive Applications*. Natick, Massachusetts: A K Peters, 2002.
- [10] G. Wang, "ChucK," 2018.
- [11] P. Cook and J. O. Smith, "Physics-Based Sound Synthesis for Games and Interactive Systems," 2018. [Online]. Available: <https://www.kadenze.com/courses/physics-based-sound-synthesis-for-games-and-interactive-systems-iv>
- [12] H. P. S. Selasky, "Evaluation of Perceptual Sound Compression with Regard to Perceived Quality and Compression Methods," Ph.D. dissertation, Høgskolen i Agder ; Agder University College, 2006.
- [13] X. Serra and J. O. Smith, "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic Plus Stochastic Decomposition," *Computer Music Journal*, vol. 14, no. 4, p. 12, 1990.
- [14] Unity Technologies, "Unity," 2018.
- [15] J. Atherton and G. Wang, "Chunity: Integrated Audio-visual Programming in Unity," in *NIME*, Virginia, US, 2018.
- [16] L. Pruvost, B. Scherrer, M. Aramaki, S. Ystad, and R. Kronland-Martinet, "Perception-Based Interactive Sound Synthesis of Morphing Solids' Interactions," in *SIGGRAPH Asia 2015 Technical Briefs*, 2015, p. 17.

# VIBRA - Technical and Artistic Issues in an Interactive Dance Project

**Andreas Bergsland**

Norwegian University of  
Science and Technology (NTNU)  
7491 Trondheim, Norway  
andreas.bergsland@ntnu.no

**Sigurd Saue**

Norwegian University of  
Science and Technology (NTNU)  
7491 Trondheim, Norway  
sigurd.saue@ntnu.no

**Pekka Stokke**

ljøs a/s  
Pir II - 13b  
7010 Trondheim, Norway  
mail@ljøs.no

## ABSTRACT

The paper presents the interactive dance project VIBRA, based on two workshops taking place in 2018. The paper presents the technical solutions applied and discusses artistic and expressive experiences. Central to the discussion is how the technical equipment, implementation and mappings to different media has affected the expressive and experiential reactions of the dancers.

## 1. INTRODUCTION

VIBRA is a project exploring expressive and artistic possibilities of interactive dance involving a group of artists based in Trondheim, Norway ([www.vibra.no](http://www.vibra.no)). The acronym VIBRA is Norwegian and translates to Video/visuals, Interaction, Movement, Space and Audio. The project started with several activities in 2017 (although, at the time, not under the name of VIBRA), involving the first and second authors and two dancers. This paper will focus on two workshops held during the spring of 2018 involving a total of eight participants.

### 1.1 Aims of the project

The main aim of the project from the beginning has been to explore interactive dance as artistic medium. Even if the project has had the artistic development at its core, it has also involved several technological components. Thus, the research questions that we aimed to answer in the project has been both of artistic and technical nature:

- How can different sensors be used in combination so as to convey movement data that corresponds well with experienced movements and how can this data be shared orderly and effectively to accommodate different mappings and media?
- How can different sensors and different mappings affect the movements of the dancers?
- How can musical and musical-spatial mappings be designed that work well for two or more dancers at a time?

It has also been important for the project to work with an inclusive conception of dance, where different bodies, abilities and levels of training are seen as productive ingredients rather than obstacles of the creative process. Several of the project participants have been working with such inclusive conceptions of dance in earlier projects [1]. For the first author, the use of sensor technologies and interactive music systems has been an important tool in expanding the embodied expressive palette of people with different abilities [2]. Openness to dialogue with, and participation from, the audience has followed from this inclusive view of dance, and has been central to our approach since the beginning.

### 1.2 Participants

Since the first collaborative projects in 2017, a total of eight people have been engaged in the project at different stages. The first author has figured as initiator, project coordinator, artistic director, programmer and composer/sound designer and been involved in all of the events. The second author has mainly been involved in the sensor communication, including programming the VIBRA-hub application. The third author made interactive computer graphics for the second VIBRA workshop and participated in the first, testing data communication and mappings. Gina Sandberg was responsible for documenting the workshops in audio and video. All project participants were considered a part of the creative team in that they could contribute in discussions and reflections happening in the workshops.

Four dancers have been involved, Arnhild Staal Pettersen, Luis della Mea, Tone Pernille Østern and Elen Øien. The three former have professional training and are also active as choreographers, while Øien is an active amateur wheelchair dancer, but has also done productions with professional dancers. All of them have danced together earlier in different projects, but have had little or no experience with interactive dance prior to the start of the project in 2017. It must also be noted that Della Mea is active both as musician and composer. Henceforth, the dancers will be referred to by the first letter of their first names (A, L, T and E).

## 2. BACKGROUND

### 2.1 Interactive dance

*Variations V* from 1965, involving John Cage, Merce Cunningham, Max Mathews, Nam June Paik and many oth-

Copyright: © 2019 Andreas Bergsland et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

ers, started a revolution when it comes to exploration of the collaboration between music, dance and technology, with dancers exerting as much influence over the sonic landscape as the musicians [3]. Following in Cage's footsteps, the term *interactive dance* has often been applied to artistic expressions in the same vein, especially as they implemented digital technology and computers from the late 1980's and onwards [4]. Mullis has defined interactive dance as "performances in which a dancer's movement, gesture, and action are read by sensory devices, translated into digital information, processed by a computer program, and rendered into output that shapes the performance environment in real time" [5]. Such environments can include media and technologies such as interactive music, lights, video, computer visuals, electro-mechanical instruments, pyro-technics, smart fabrics, internet interfaces and more, in combination with non-interactive elements or not [4–10].

A particular trait of interactive dance is that it makes little sense to apply a fixed choreography in the creation process. Without some freedom for the dancers to go beyond a strict choreography, e.g. moving freely or using structured improvisation, the interactive system will produce a more or less fixed output [6, 11]. Thus, interactivity can be seen as something that increases the creative control of dancers [12] and gives them more freedom of expression, especially in the temporal domain [4].

### 3. OUR APPROACH

While there have been many technological challenges to the VIBRA project, the core of it has been to explore and develop the artistic possibilities and experiences of interactive dance. In that it aimed to generate knowledge through an embodied and situated artistic practice embedded in artistic and academic contexts, our approach has affinities with Borgdorff's conception of *artistic research* [13]. However, since we consider our workshops more as work-in-progress, and since we haven't at this point conducted a deeper analysis and discussion of them, we don't regard our work as a completed artistic research project.

Our work also has affinities with approaches such as *practice-as-research*, *practice-based* or *practice-led* research [14]. In making the case for the latter of these, Grey discusses the role of what she calls the "practitioner-researcher", and this role seems highly fitting for the first author's role in this project: "The role is multifaceted - sometimes generator of the research material - art/design works, and participant in the creative process; sometimes self-observer through reflection on action and in action, and through discussion with others; sometimes observer of others for placing the research in context, and gaining other perspectives; sometimes co-researcher, facilitator and research manager, especially of a collaborative project" [15]. Ideally, the reflective part of the research process could have been made epistemologically more robust by inviting all project participants to observe and reflect on the documentation of the workshops, but this was unfortunately not practically possible at this point. The dancers' experiences and reflections are therefore implied from analysis of the video mate-

rial, where the alternation between dialogue/reflection and practical exploration still attests to the importance of reflexivity, dialogue and common artistic development throughout the workshops.

Finally, it is not difficult to argue that the kind of knowledge production Borgdorff refers to above is processual and *embodied* [16]. It is only by moving, listening, feeling one's own body and observing others' that one can develop knowledge of the most important aspects of interactive dance. Therefore, it has always been important for the project to let all project participants experience the interaction.

## 4. TECHNICAL SETUP AND VENUES

### 4.1 Venues, workshop structure and documentation

The first workshop took place over two days at DansIT (<http://www.dansit.no/>), a dance studio owned by a network organization for dance in the region going under the same name. The workshop was conducted in a very open and exploratory manner, where the focus was on including all the participants in a collective creative process developing material for a showing at the end of each workshop. In the process, the dancers got to test and respond to different sensors and mappings, and then develop movement material through improvisation. The structuring of this material for the showings was done collectively. We also made room for conversations during and at the end of each workshop day, so as to sum up the salient experiences of all participants. The experiences and reflections then guided the further development of instruments and mappings, which happened before and in-between the workshop hours.

For the first workshop, the focus was on the audio part of the interaction, although the third author was present and making tests of his setup for computer visuals. Due to the available equipment at the venue, and that the technical setup required quite a bit of setup time, we used the available stereo PA system at the venue.

The second workshop took place at Verkstedhallen, a black box theatre in Trondheim (<http://www.verkstedhallen.no/>). In addition to the mentioned sensors and the action-sound mappings, this workshop also integrated computer visuals and 8-channel spatial audio (See fig.1).

Both workshops, were recorded in audio and video by the videographer, Gina Sandberg, using two cameras; one on a stand, and the other handheld. The audio and video documentation, along with notes and computer files applied in the technical setup, form the material that is the basis for this paper.

### 4.2 Sensors

In the two workshops discussed, we have explored two types of sensors:

1. NGIMU sensors, 9DOF IMU sensors with on-board sensor fusion algorithms for absolute orientation
2. Myo armbands, combining 8 EMG electrodes and 9DOF IMU sensor

These were chosen because they were lightweight and robust, they were easy to fasten on the dancers and had wireless communication with low latency. In addition, they both supported, directly or indirectly via available and easy-to-use software, the OSC-protocol, something which greatly facilitated the data communication (see sect. 4.3).

#### 4.2.1 NGIMU

The NGIMU sensor<sup>1</sup> communicates with OSC-messages over Wi-Fi networks, both for transmitting sensor data and for configuration. The use of Wi-Fi technology allows higher data rates, longer operating range and higher transmission power at the cost of higher power consumption and shorter battery life compared to competing technologies such as ZigBee and Bluetooth [17].

The AHRS sensor fusion algorithm provides several measures of absolute orientation, but we focused on the Euler angles (roll, pitch and yaw) that presented stable results in most cases. A noticeable exception was the case when pitch approached  $\pm 90^\circ$  leading to erratic measures of roll and yaw (the gimbal lock problem). In order to stabilize the orientation measures we discarded incoming measures of roll and yaw when pitch approached the critical angle.

Another problematic issue appears when the sensor is rotated. Both roll and yaw display discontinuous jumps between  $-180^\circ$  and  $+180^\circ$ . In most practical cases it is sufficient to unwrap the angle values by adding or subtracting  $360^\circ$  when a jump is detected (in our case defined as a value change exceeding a threshold of  $320^\circ$ ). It is not a permanent solution, but it holds for the typical movement patterns of our dancers.

#### 4.2.2 Myo armbands

The Myo armband<sup>2</sup>, with its combination of EMG and IMU sensors in one compact wearable armband, has been explored both as a DMI [18] and also more relevantly here, in dance contexts [10, 19–21]. The Myo armband communicates over Bluetooth LE, but several software tools are available for mapping Myo data into OSC. We chose Myo Mapper that incorporates several useful functions for filtering, scaling, calibration and error correction [22]. One Myo Mapper instance can communicate with only one armband, but up to three instances may run on a single computer.

### 4.3 Data Communication

The technical setup involved three computers in order to distribute tasks, responsibilities and load among the three authors (see Fig. 1):

1. First author controlled the interactive instruments and spatialized sound output.
2. Second author controlled sensor communication and data distribution.
3. Third author controlled computer visuals.

<sup>1</sup> <http://x-io.co.uk/ngimu/>

<sup>2</sup> <https://support.getmyo.com/>. The product has been discontinued since Oct 2018.

The computers were interconnected through a router that also served as a wireless access point for the NGIMU sensors. The three computers all needed access to the same sensor data, but the sensors communicated exclusively with a single host. Hence, we decided to establish a hub for data distribution on the second computer, the *VIBRAhub* (See fig.1). This is an application written specifically for this purpose.<sup>3</sup> OSC-messages sent from the sensors to the hub are immediately passed on to all connected receivers, but with a unique address prefix added to identify each sensor. We have not implemented message filtering at this stage.

The *VIBRAhub* communicated directly with all NGIMU-sensors. It also took care of the Euler angle issues discussed in section 4.2.1 before passing the data on. We also found it useful to monitor the battery status of the NGIMU sensors in the *VIBRAhub* graphical interface, as a reminder to recharge when required during a long workshop.

The Myo armbands communicated over Bluetooth to their designated Myo mapper, which translated the data to OSC messages and passed them on to the *VIBRAhub*. Due to the limitation of three Myo mappers on a single computer, one of the four Myo armbands had to connect to a different computer. Nevertheless, all Myo messages were sent to the *VIBRAhub* and redistributed from there to the connected receivers.

### 4.4 Interactive instruments and Mappings

All of the interactive instruments were programmed using Csound.<sup>4</sup> The data from the VIBRA-hub were received via an Ethernet connection for minimal latency, using the OSC implementation in Csound (OSClisten opcode). The setup in Csound was flexible in that data from all sensors could be routed to the different instruments, and also in that every instrument could be set up with different parameters to create variations in their sonic qualities. However, due to the particular nature and structure of the EMG data, some instruments were designed particularly for the data they provided. Four simple instrument sketches were developed prior to the first workshop, whereas the majority of the instrument development happened in response to what happened in the workshops. A total of eight instruments were used in the workshops, where five used different methods of sound synthesis and three used different kinds of processing of sampled sounds. The interactive instruments and their mappings have been described in more detail in a blog post on the *vibra.no* website, including video examples.<sup>5</sup>

The instruments also varied with regards to which sonic parameters that were modulated and the degree of temporal synchronization in relation to the input data. The instruments we called *MultiSine* (used with the Myo EMG) and *Noizer* (used with the NGIMU) were perhaps the most unclear and clear with regards to causality, respectively. For the *MultiSine* instrument this was due to using the values from the EMGs without any gating, so that the base tension

<sup>3</sup> Available as open source at <https://github.com/ssaue/Vibra>

<sup>4</sup> <https://csound.com>

<sup>5</sup> <https://www.vibra.no/blogg/interactive-instruments-and-mappings-used-in-the-vibra-workshops>



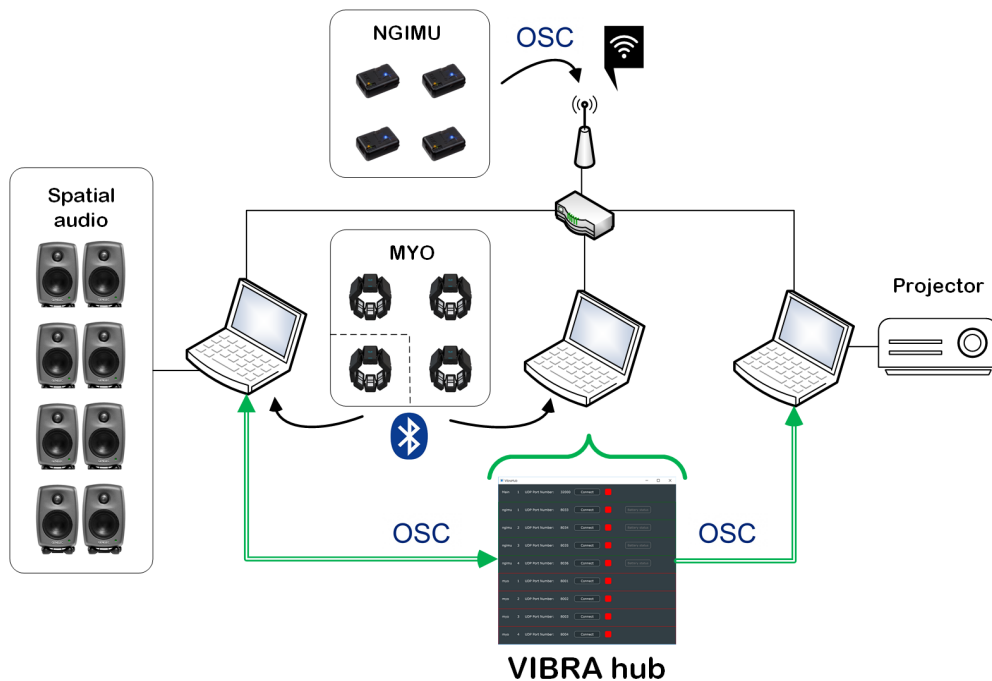


Figure 1. The VIBRA setup. A central computer communicates with NGIMU and Myo sensors and distributes sensor data to the computers controlling sound and visuals.

in the muscle would produce sound, and that the triggering of a new sine tone would have to wait until the already playing tone was finished.

#### 4.5 Sound Spatialization

In the second workshop we used a 8-channel speaker setup with eight Genelec 8030A speakers in a circle configuration around the dancing area. The audience was localized along the rim of this circle, but we were careful not to place anybody in front of any loudspeaker. In this way, the audience would not experience any surround sound, but rather a complex spatial image in front and to their sides. Moreover, this spatial image would differ depending on the placement of the individual spectator.

The sound spatialization at the second VIBRA workshop was implemented using IRCAM Spat running in Max [23], with spatialization parameters sent from Csound to Max internally in the computer using the `csound~` object in Max. We used the vector based amplitude panning as implemented in the `spat.spat~` object with the `vbap2D` panning method with 8 individual sources. This enables control of azimuth, distance and spread for each source, and offers many settings to adjust the audible characteristics of the simulated reverberation.

We tested a number of mappings between the sensor data input and the spatialization parameters through a flexible user-defined-opcode in Csound, where we could choose mappings and parameter settings. Having tried out a relatively straightforward mapping in which the direction of the torso of the dancer controlled the azimuth angle of the

spatial image at an earlier collaboration<sup>6</sup>, we wanted this time to use a less direct mapping with more movement and variation. The gyro values of the NGIMU and Myo sensors were used to get the change in angle of the body part wearing it (arm and calf), and we tested how this could be used to control the location, the speed and direction of rotation, as well as the distance of the individual sound sources.

All in all, the spatialization added spatial dynamics and interest, as well as clarified the source separation for each of the dancers. Since it perhaps was the least developed part of the performance, it will not be discussed in further detail in this paper.

#### 4.6 Computer Projections

The computer projections were set up for the second of the VIBRA workshops, and were controlled partly through the use of sensor data from the dancers, and partly manually controlled by the third author. As for the former part of the technical setup, it consisted of a simple receiver matrix to accept all sensor data coming out of the VIBRA hub controller system. The incoming data was then sorted and smoothed, before being routed to Processing<sup>7</sup> and VDMX<sup>8</sup> running shaders and FFGL plugins, using OSCulator<sup>9</sup> for internal data communication. Within these environments the data was mapped to triggers for different actions, large and small, generative and affective. As for the manual control, the third author would observe the dancers' performance with his fingers on the faders of his

<sup>6</sup> <http://folk.ntnu.no/andbe/Sound-Space-Movement>

<sup>7</sup> <https://processing.org>

<sup>8</sup> <https://vidvox.net>

<sup>9</sup> <https://osculator.net>

MIDI controller hooked up to his computer, adjusting the routing of sensor data according to his own aesthetic preferences. The computer visuals were projected onto a black stage carpet hung on one of the walls in the black box using a Panasonic 10 000 lumen projector.

## 5. ARTISTIC EXPERIENCES AND REFLECTIONS

After the technical setup in each workshop, the structure was quite open and exploratory. Besides simple explanations of the functionality of the two sensors and some help of attaching them to their bodies, the dancers had no specific instructions or choreography. However, since all of the dancers had extensive experience in improvising, the implicit expectations were that they improvised freely within the interactive system. With an open showing at the end of each workshop, it was also implied that the group developed some material to be presented there.

### 5.1 Experiences of Individual Causal Relationships

The initial focus for the dancers after putting on the sensors and starting the interactive instruments was to establish causal relationships between their individual movements and the sonic feedback it caused. All of the dancers intuitively started to move the limb with the sensor and listened to identify their individual sounds. However, in those cases where the initial instruments were relatively similar in sound quality, they had some problems discerning which dancer made which sound. After a suggestion from one of the dancers, they decided to dance only two at a time, to make the causal relationships between movement and sound clearer.

However, after the initial causal relations had been established, the nature of the relationships was something that both interested dancers as well as audience. The dancers would express how they found *variation* in the degree of directness of the mappings interesting. And, during the de-brief after the showing at the first workshop, one of the spectators observed how the time between movement and sound differed between the different instruments, and found it exciting to ponder which movement caused which sound.

### 5.2 Placement of Sensors Affected Movements

In several of the sessions during both workshops it was interesting to observe the degree to which the positioning of especially the Myo sensor seemed to have an affect on how the dancers moved. In the first workshop, three of the dancers tried to have the Myo sensor positioned on their calf.

L in particular seemed to be very conscious of how he could activate the muscles in his calf to affect the sensor values: He often put the weight on the leg not wearing the sensor, and then carefully stretching out the sensor leg towards the floor and pushing the heel up and down. Frequently, he put all his weight on the leg with the sensor, seemingly to achieve a peak value. At other times, he lifted it the sensor leg from the floor, thereby minimizing muscle activation and consequently reducing the auditory feed-

back to the minimal. With the foot lifted from the ground, he would also rotate, bend and stretch the foot, as well as keeping it completely still and kicking forcefully into the air. In this position, naturally, the muscle activation would be freed from efforts of standing or putting weight on the foot. Generally, it was notable how L in this phase of the workshop appeared to act as a musician "playing" an instrument.<sup>10</sup>

It was also interesting to notice how T's improvisations were quite different when she wore the sensor on her arm compared to when she wore it on her calf. For instance, she frequently emphasized the movement or the position of the limb wearing the sensor through either following it with her eyes, keeping the rest of her body still while moving it, moving it more, moving it with a more emphatic quality (often stretched out/erected) than the rest of her body, or positioning it with some distance from the rest of her body. This emphasis appeared quite differently with the sensor on the lower arm than on the calf. The same type of emphasis was also apparent with the NGIMU sensors, but since we just tried localising it in the hand or on the lower arm, the difference was not so striking.

Lastly, it could sometimes be observed how, especially when wearing only one sensor (most of the time in the second workshop they would wear two), the dancers were able to clearly separate the sound playing body part from the rest of the "dancing" body. In particular A, when wearing one Myo armband, would sometimes relax the muscles in her sensor arm, while moving the rest of her body, and then, in response to that, initiate a movement with the sensor arm activating the muscle and thereby the sonic response. Thus, she seemed to engage in a creative dialogue between the "playing" and "dancing" parts of her body.

### 5.3 Playing Gestures are Included in the Repertoire

An analysis of the movements of the dancers during the workshop also showed that many of the types of movements that the dancers initially used to explore the sonic affordances of the sensors on one of their limbs, were later also applied in the body parts not wearing sensors as a part of the improvisation repertoire. In a sense, the playing gestures were treated like musical motifs first being introduced with a focus on the interaction with the sound, and subsequently taken up and developed further in interplay with body parts without sensors and other dancers.

One example was a duet with T and A in the first day of the first workshop. Here, T in her initial exploration of the sonic affordance space of the Myo armband started to push her arm towards the floor to get a sonic response. After T had explored several other ways of heightening the sonic response through touching the floor - pushing, sweeping, stroking - A then took up the motif by standing on her knees and hands, and then crawled over the floor, alternately pushing both hands emphatically towards the floor in a rhythmical manner - simultaneously creating a clearly audible pulse. The motif was further emphasized when T then picked up the rhythm by repeatedly pushing her right

<sup>10</sup> It might be that his background as a musician and composer has affected his approach to the interactive instruments.



Figure 2. Tone (left) and Arnhild (right) synchronously pushing towards the floor.

hand towards the floor while moving in the opposite direction (see Fig.2). The simultaneous interplay both movement and sound made this a particularly heightened point in the improvisation.<sup>11</sup>

#### 5.4 Character of Instruments Affected Movements

The instruments used with the different sensors employed in the two workshops were quite different in character. Some of the instrument had a relatively tight coupling between the effort or energy in the movement or muscle activation and the intensity and character of the sound, whereas others had a more indirect coupling. It could be observed that the more direct couplings often tended to motivate faster and more abrupt movements with more effort, whereas the latter tended to have more flow and less effort.

One instrument, which we called the "Noizer", stood out in that respect. The instrument featured four layers of differently modulated and filtered noise, rendering a relatively complex texture. The intensity and character of the noise was mapped to a normalized acceleration vector, using an algorithm for intensity ported from the IRCAM's RIoT-intensity Max object<sup>12</sup> to Csound. The quite immediate coupling between the energy/effort of the movements and the intensity of the sound appeared to often instigate movements with high effort, speed, energy and/or abruptness in the dancer's improvisations. Interestingly, this could also be observed during the audience's tryout of the sensors after the first workshop.

#### 5.5 Artistic Expression through Chairs

Dancing in a wheelchair naturally presents different affordances for movement and expression in space, time and

dynamics than other forms of dance [1,24]. While E definitely had an ample expressive possibilities with the Myo and NGIMU attached to her lower arms in the first workshop, her arms were nevertheless restricted by the need of navigating the wheelchair in the dance quite frequently. At the beginning of the second workshop, we wanted to better capture salient features of wheelchair dance and thus extend E's expressive possibilities. To do this, we kept the Myo on E's arm, but also attached a NGIMU sensor on one of the wheels. By using the delta of the orientation we got a value correlating with the speed of the wheel. Playing the instrument called *Bipp-a-chu*, this value was mapped to the frequency of an impulse generator which modulated continuous sound samples, producing a sort of sonic analogy to the ticking of spokes in a wheel. With these new possibilities E started moving the wheelchair a lot more and with higher speed and with more pirouettes than during the first workshop. She also started to explore going up on the back wheels and balancing there, something which was eventually included in the final showing before the audience at the end of the workshop.

The extended expressive possibilities of the wheelchair also spurred an idea to also let T use a chair in her dance, and then make it into a duet with two chairs; one with legs and one with wheels. We attached a NGIMU sensor to one of the legs of the chair, and used it to play the *Noizer* instrument discussed above. T could then play the chair by moving it around. Often she played the chair by lifting it into the air, rotating it and swinging it from side to side. While the chair as an object to sit in normally is a passive and stationary object, its role in this context was radically changed and estranged. Although this "Noizer-chair" both had very different sound and movement affordances than the wheelchair, the fact that both were objects to sit in, created an interesting conceptual link that gave an interesting artistic perspective, inviting the audience to reflect on what chairs can be.

#### 5.6 Aesthetic Considerations of the Computer Visuals

The computer visuals were in the end perhaps assigned a less salient function in the interactive experience than the audio part, with which the dancers had worked the entire first workshop. Projected on a black stage carpet covering one of the walls in the performance area of the second workshop, it provided mainly a visual accompaniment to the dancers, who still from time to time were projected upon when they approached the wall. The role of the visuals were still considered important, in that it provided the performance with an additional aesthetic element that framed and interacted with dancers movements, especially when they were projected upon.

Mapping a lot of the raw sensor data directly to the computer visuals would create a very direct correlation between cause and effect, which can in some cases be considered too "obvious" and perhaps of limited aesthetic interest. Instead, the third author's approach was based on an analysis of the different dancers' movement strategies, especially the sensor data with slower changes in the values, and this was combined with manual routing to parameters of the

<sup>11</sup> This can be seen in the video demonstrating the WarbleSine instrument at <https://youtu.be/jrh7-PXjVdk>

<sup>12</sup> <http://forumnet.ircam.fr/product/bitalino-r-iot/>

graphics software. In this way, the visualizations could take a more over-arching role in relation to the movements - sometimes working more in a contrapuntal rather than mimicking fashion - depending on the third author's subjective aesthetic preferences and emotional response.

## 6. DISCUSSION

During the two workshops we had a clear experience that when working with dancers controlling sensor-based musical instruments, the sense of clear causality was an important factor. Especially when starting to work with a sensor and a particular instrument, it was important for them to establish who made what sound. For the computer visuals, however, it was a point that the relationship between the projected material and the movement of the dancers was not too direct, even though the data from the sensors was actually affecting the output.

This touches a central aesthetic point in interactive dance, and interactive art in general, namely the *experience* of correlation, correspondence, similarity or causality between the input (here: the dancer's movement) and the output (here: the computer visuals and the sound) [25]. There are several terms for this in the writings about interactive dance: Mullis calls it "birectionality", Wilson and Bromwich uses the term "awareness", and Rizzo and colleagues refers to it as "feedback" [5, 10, 12]. No matter which term is used, it is important to focus on the *experiential* aspects of this relationship rather than the factual, both from the audience and performers' perspectives. On one side, there can exist a factual coupling between sensor data and the sonic or visual output even if it can't be perceived, and on the other side, the dancer or audience member can experience a causal relationship out of pure coincidence. One could see this as translatable to a continuum from very clearly experienced causality to the complete absence of experienced causality, e.g. by being random, asynchronous or nonexistent [4]. These poles can also be linked to aesthetic judgments where clear causality might be seen as "simplistic", "banal" or "naïve", and the opposite pole as "opaque" or "inaccessible". However, the latter pole can also be seen as undermining the very idea of interactivity, in that it inhibits the reciprocity of cause-effect that is implied with interaction.<sup>13</sup> This also mirrors different aesthetic approaches within choreography, with choreographers like Lopukhov on one side, arguing for a "complete union between dance and music" (Lopukhov 2002, 142, cited in [26]), and Cunningham, who in 1952 argued for the individual autonomy of dance on one side, and music on the other [27].

Not surprisingly, we could observe in our workshops how both the type of sensor, its placement on the dancer's body and the instruments used affected the dancers' movements, and thus that the "allowance", "bidirectionality" or "feedback" definitely was an active and most often observable component of the interaction. What was interesting, was that the movements that were directly related to the interaction entered into the improvisational movement reper-

<sup>13</sup> Admittedly, and often a case in interactive art, providing the audience with an explication of the causal relationships through liner notes, programmes, etc. can also affect the experience of correlation.

toire, both for each individual dancer, but also in between dancers, thus creating a dialogue between the active/controlling and the passive/contextual movements - what Wilson and Bromwich refer to as "online" and "offline", respectively [12].

At times, the separation of the body parts "playing" and "dancing" appeared to allow for a more marked division of "offline" and "online" body parts, and this was then actively used in creative interplay. When we placed the sensor on the chair that T was playing with, her "playing" and "dancing" could naturally be completely separated if needed. Conversely, when E danced with a sensor on her wheelchair, she *had to* make sound whenever she moved it. Hence, issues of control, empowerment, freedom and coercion are all at stake in interactive dance.

## 7. CONCLUSIONS AND FUTURE WORK

During the two VIBRA workshops, we experienced a process in which four dancers familiarized themselves with interactive technology that eventually enabled them to let their dance movements affect spatialized musical sound and computer visuals in a performance. The workshops highlighted issues related to causality and interactivity, and how these can be differently expressed along a continuum from clear to opaque causal relationships. Moreover, we saw how "dancing" and "playing" could function as more or less independent components in the interactive dance expression.

Lastly, even if the long term goals have been to develop artistic productions, the focus so far has been on developing technological solutions for complex setups and media mappings with multiple performers. Moreover, rather than working from preconceived artistic ideas, we have emphasized exploration of the artistic possibilities related to the technical materialities of these setups. Thus, we see the discussed workshops as stepping stones towards more developed artistic productions intended for public presentation in the future.

## Acknowledgments

The VIBRA project was supported by NTNU (ARTEC, The Dept. of Music and The Dept. of Teacher Education), The Arts Council Norway, DansIT, Trondheim kommune, Trøndelag fylke and Fond for utøvende kunstnere.

## 8. REFERENCES

- [1] T. P. Østern and E. Øien, "Moving through change," *Choreographic Practices*, vol. 6, no. 1, pp. 125–144, 2015.
- [2] A. Bergsland and R. Wechsler, "Turning movement into music: Issues and applications of the MotionComposer, a therapeutic device for persons with different abilities," *SoundEffects-An Interdisciplinary Journal of Sound and Sound Experience*, vol. 6, no. 1, pp. 23–47, 2016.



- [3] L. E. Miller, "Cage, Cunningham, and Collaborators: The Odyssey of 'Variations V'," *The Musical Quarterly*, vol. 85, no. 3, pp. 545–567, 2001. [Online]. Available: <http://www.jstor.org/stable/3600996>
- [4] W. Siegel, *Dancing the Music: Interactive Dance and Music*. Oxford: Oxford University Press, 2009, ch. 10, pp. 191–213.
- [5] E. Mullis, "Dance, interactive technology, and the device paradigm," *Dance Research Journal*, vol. 45, no. 03, pp. 111–123, 2013. [Online]. Available: <http://dx.doi.org/10.1017/S0149767712000290>
- [6] J. Birringer, "Interactive dance, the body and the internet," *Journal of Visual Art Practice*, vol. 3, no. 3, 2004.
- [7] T. Todoroff, "Wireless digital/analog sensors for music and dance performances," in *Proceedings of the International Conference on New Interfaces of Musical Expression, NIME'11*, 2011, pp. 515–518.
- [8] D. Bisig, P. Palacio, and M. Romero, "Piano & dancer," in *19th Generative Art Conference GA2016*, 2016, pp. 138–154.
- [9] K. Woolford and C. Guedes, "Particulate matters: Generating particle flows from human movement," in *ACM International Conference on Multimedia*, 2007, pp. 691–696.
- [10] A. Rizzo, K. El Raheb, S. Whatley, R. M. Cisneros, M. Zanon, A. Camurri, V. Viro, J.-M. Matos, S. Piana, and M. Buccoli, "WhoLoDancE: Whole-body Interaction Learning for Dance Education," in *Proceedings of the Workshop on Cultural Informatics (co-located with the EUROMED 2018)*, A. Angeliki and W. Manolis, Eds., 2018, pp. 41–50.
- [11] J. James, T. Ingalls, G. Qian, L. Olsen, D. Whiteley, S. Wong, and T. Rikakis, "Movement-based interactive dance performance," in *Proceedings of the 14th ACM international conference on Multimedia*. ACM, 2006, pp. 470–480.
- [12] J. A. Wilson and M. A. Bromwich, "Lifting bodies: interactive dance – finding new methodologies in the motifs prompted by new technology – a critique and progress report with particular reference to the Body-coder system," *Organised Sound*, vol. 5, no. 1, pp. 9–16, 2000.
- [13] H. Borgdorff, *The Production of Knowledge in Artistic Research*. London: Routledge, 2011, ch. 3, pp. 44–63.
- [14] B. C. Haseman, "A manifesto for performative research," *Media International Australia Incorporating Culture and Policy: Quarterly journal of media research and resources*, no. 118, pp. 98–106, 2006. [Online]. Available: <http://eprints.qut.edu.au/3999/>
- [15] C. Grey, *Inquiry through practice: developing appropriate research strategies*. Helsinki: Research Institute, University of Art and Design, Helsinki UIAH, 1998, pp. 82–95.
- [16] M. Johnson, *Embodied knowing through art*. London: Routledge, 2011, ch. 8, pp. 141–151.
- [17] T. Mitchell, S. Madgwick, S. Rankine, G. S. Hilton, A. Freed, and A. R. Nix, "Making the most of wi-fi: Optimisations for robust wireless live music performance," in *Proceedings of the International Conference on New Interfaces of Musical Expression, NIME'11*, 2014, pp. 251–256.
- [18] K. Nymoen, M. R. Haugen, and A. R. Jensenius, "Mumyo – evaluating and exploring the myo armband for musical interaction," in *Proceedings of The International Conference on New Interfaces of Musical Expression Conference*, 2015, pp. 215–219.
- [19] I. Jarvis and D. V. Nort, "Posthuman gesture," in *Proceedings of the 5th International Conference on Movement and Computing*. 3212807: ACM, 2018, pp. 1–8.
- [20] U. Côté-Allard, D. St-Onge, P. Giguère, F. Laviolette, and B. Gosselin, "Towards the use of consumer-grade electromyographic armbands for interactive, artistic robotics performances," in *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2017, pp. 1030–1036.
- [21] J. F. Houser, "Reflections: For interactive electronics, dancer, and variable instruments," Dissertation in Fine Arts, 2014. [Online]. Available: <http://hdl.handle.net/2346/58674>
- [22] B. Di Donato, J. Bullock, and A. Tanaka, "Myo Mapper: a Myo armband to OSC mapper," in *Proceedings of the International Conference on New Interfaces of Musical Expression*, 2018, pp. 138–143.
- [23] T. Carpentier, "Une nouvelle implémentation du Spatialisateur dans Max," in *Journées d'Informatique Musicale (JIM 2018)*, L. Bigo, M. Giraud, R. Groult, and F. Levé, Eds., vol. May, 2018, 2018, pp. 45–52. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-01791435>
- [24] S. Whatley, "Dance and disability: the dancer, the viewer and the presumption of difference," *Research in Dance Education*, vol. 8, no. 1, pp. 5–25, 2007. [Online]. Available: <https://doi.org/10.1080/14647890701272639>
- [25] F. Berthaut, D. Coyle, J. W. Moore, and H. Limerick, "Liveness through the lens of agency and causality," in *The 15th International Conference on New Interfaces for Musical Expression, NIME'15*, 2015.
- [26] P. H. Mason, "Music, dance and the total art work: choreomusicology in theory and practice," *Research in Dance Education*, vol. 13, no. 1, pp. 5–24, 2012. [Online]. Available: <https://doi.org/10.1080/14647893.2011.651116>
- [27] M. Cunningham, *Space, time and dance*. New York: Da Capo Press, 1998, pp. 37–39.

# Musical Tempo and Key Estimation using Convolutional Neural Networks with Directional Filters

**Hendrik Schreiber**

tagtraum industries incorporated  
hs@tagtraum.com

**Meinard Müller**

International Audio Laboratories Erlangen  
meinard.mueller@audiolabs-erlangen.de

## ABSTRACT

In this article we explore how the different semantics of spectrograms' time and frequency axes can be exploited for musical tempo and key estimation using Convolutional Neural Networks (CNN). By addressing both tasks with the same network architectures ranging from shallow, domain-specific approaches to deep variants with directional filters, we show that axis-aligned architectures perform similarly well as common VGG-style networks developed for computer vision, while being less vulnerable to confounding factors and requiring fewer model parameters.

## 1. INTRODUCTION

In recent years Convolutional Neural Networks (CNN) have been employed for various Music Information Retrieval (MIR) tasks, such as key detection [1, 2], tempo estimation [3], beat and rhythm analysis [4–6], genre recognition [7, 8], and general-purpose tagging [9, 10]. Typically, a spectrogram is fed to the CNN and then classified in a way appropriate for the task. In contrast to recent computer vision approaches like Oxford's Visual Geometry Group's (VGG) deep image recognition network [17], some of the employed CNN architectures for MIR tasks use rectangular instead of square filters. The underlying idea is that, while for images the axes *width* and *height* have the same meaning, the spectrogram axes *frequency* and *time* have fundamentally different meaning. For MIR tasks mainly concerned with temporal aspects of music (e.g., tempo estimation, rhythmic patterns), rectangular filters aligned with the time axis appear suitable [3]. Correspondingly, tasks primarily concerned with frequency content (e.g., chord or key detection), may be approached with rectangular filters aligned with the frequency axis [11]. In fact, tempo and key estimation can be seen as tasks from two different ends of a spectrum of common MIR tasks, which are addressed by systems relying more or less on temporal or spectral signal properties (Figure 1). Systems for other tasks like general-purpose tagging or genre recognition are found more towards the center of this spectrum as they usually require

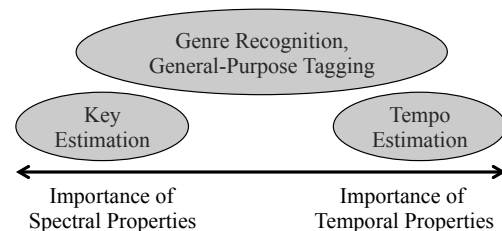


Figure 1: Several MIR tasks and their reliance on spectral or temporal signal properties.

both spectral and temporal information.

In [12] Pons et al. explored the role of CNN filter shapes for MIR tasks. In particular, they examined using rectangular filters in shallow CNNs for automatic genre recognition of ballroom tracks. Defining *temporal* filter shapes as  $1 \times n$  and *spectral* filter shapes as  $m \times 1$ , they showed that using temporal filters alone, 81.8% accuracy can be reached, which is in line with a Nearest Neighbour classifier (k-NN) using tempo as feature scoring 82.3% [13]. Using just spectral filters, the test network reached 59.6% accuracy, and a fusion architecture with both temporal and spectral filters performed as well as an architecture using square filters, scoring 87%. The experiments confirmed that such *directional* filters can be used to match either temporal or spectral signal properties and that both may be useful for genre recognition.

Even though directional filters did not outperform square filters, there are good arguments for using them: First, CNNs using specialized, directional filters may use fewer parameters or match musical concepts using fewer layers [14]. Second, by limiting what a filter can match, one can influence what a CNN might learn, thus better avoid “horses” [15] and improve explainability. The latter is especially interesting for genre recognition systems, given their somewhat troubled history with respect to explicit matching of musical concepts [14, 16]. To further explore how and why directional or square filters contribute to results achieved by CNN-based classification systems for MIR tasks, we believe it is beneficial to build on Pons et al.’s work and experiment with tasks that explicitly aim to recognize either high-level temporal or spectral properties, avoiding hard to define concepts like genre. Such tasks are global key and tempo estimation.

The remainder of this paper is structured as follows: In Section 2 we describe our experiments by defining both tasks, the used network variants, the training procedure,

Copyright: © 2019 Hendrik Schreiber et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

and evaluation. The results are then presented in Section 3 and discussed in Section 4. Finally, in Section 5 we present our conclusions.

## 2. EXPERIMENTS

For the purpose of comparing the effects of using different filter shapes we train and evaluate different CNN architectures for the key and tempo estimation tasks using several datasets. In this section, we first describe the two tasks, then discuss the used network architectures and datasets, and finally outline the evaluation procedure.

### 2.1 Key Estimation

Key estimation attempts to predict the correct key for a given piece of music. Oftentimes, the problem is restricted to major and minor modes, ignoring other possible modes like Dorian or Lydian, and to pieces without modulation. Framed this way, key estimation is a classification problem with  $N_K = 24$  different classes (12 tonics, major/minor). The current state-of-the-art system is CNN-based using a VGG-style architecture with square filters [2] and a fully convolutional classification stage, as opposed to a fully connected one. This allows training on short and prediction on variable length spectrograms.

In our experiments we follow a similar approach. As input to the network (Section 2.3) we use constant-Q magnitude spectrograms with the dimensions  $F_K \times T_K = 168 \times 60$ ;  $F_K$  being the number of frequency bins and  $T_K$  the number of time frames.  $F_K$  covers the frequency range of 7 octaves with a frequency resolution of two bins per semitone. Time resolution is 0.19 s per time frame, i.e. 60 frames correspond to 11.1 s. Since all training samples are longer than 11.1 s, we choose a random offset for each sample during each training epoch and crop the spectrogram to 60 frames. To account for class imbalances within the two modes, each spectrogram is randomly shifted along the frequency axis by  $\{-4, -3, \dots, 6, 7\}$  semitones and the ground truth labels are adjusted accordingly. We define *no shift* to correspond to a spectrogram covering the 7 octaves starting at pitch E1. In practice, we simply crop an 8 octaves spanning spectrogram that starts at C1 to 7 octaves. After cropping the spectrogram is normalized so that it has zero mean and unit variance.

### 2.2 Tempo Estimation

Even though tempo estimation naturally appears to be a regression task, Schreiber and Müller [3] have shown that it can also be treated as a classification task by mapping Beats Per Minute (BPM) values to distinct tempo classes. Concretely, their system maps the integer tempo values  $\{30, \dots, 285\}$  to  $N_T = 256$  classes. As input to a CNN with temporal filters and elements from [18] and [14] they use mel-magnitude-spectrograms. Even though we work with other network architectures than [3] (Section 2.3), we use the same general setup. We also treat tempo estimation as classification into 256 classes and use mel-magnitude-spectrograms with the dimensions  $F_T \times T_T = 40 \times 256$  as input;  $F_T$  being the number of frequency bins and  $T_T$

Module	Module	Size
ShallowMod	DeepMod	$\ell = 0$
	DeepMod	$\ell = 1$
	DeepMod	$\ell = 2$
	DeepMod	$\ell = 2$
	DeepMod	$\ell = 3$
	DeepMod	$\ell = 3$
ClassMod	ClassMod	
(a) Shallow	(b) Deep	

Table 1: Used network architectures. (a) Shallow architecture consisting of a variant of the ShallowMod module and a ClassMod module. (b) Deep architecture consisting of multiple, DeepMod modules parameterized with  $\ell$  to influence the filter count and a ClassMod module.

the number of time frames.  $F_T$  covers the frequency range 20 – 5,000 Hz. The time resolution is 0.46 ms per time frame, i.e., 256 frames correspond to 11.9 s.

Just like the training excerpts for key estimation, the mel-spectrograms are cropped to the right size using a different randomly chosen offset during each epoch. To augment the training dataset, spectrograms are scaled along the time axis before cropping using the factors  $\{0.8, 0.84, \dots, 1.16, 1.2\}$ . Ground truth labels are adjusted accordingly [3]. After cropping and scaling spectrograms are normalized ensuring zero mean and unit variance per sample.

### 2.3 Network Architectures

To gain insights into how filter shapes affect performance of CNN-based key and tempo estimation systems we run experiments with two very different architectures: a relatively shallow but specialized one, and a commonly used much deeper one from the field of computer vision. Both architectures are used for both tasks.

#### 2.3.1 Shallow Architectures

Our Shallow architectures, outlined in Table 1a, consists of two parts: the feature extraction module ShallowMod and the classification module ClassMod. ShallowMod, depicted in Table 2a, is inspired by a classic signal processing approach that first attempts to find local spectrogram peaks along one axis, averages these peaks over the other axis, and then attempts to find a global pattern, i.e., a periodicity for tempo estimation [19] and a pitch profile for key detection [20]. In terms of CNNs this means that our first convolutional layer consists of short directional filters (local peaks), followed by a one-dimensional average pooling layer that is orthogonal to the short filters, followed by a layer with long directional filters (global pattern) that stretch in the same direction as the short filters. We use ReLU as activation function for the convolutional layers and to avoid overfitting we add a dropout layer [21] after each ReLU. The parameters  $k$  and  $p_D$  let us scale the number of convolutional filters and dropout probabilities.

(a) ShallowMod			
Layer	Temp	Spec	Square
Input			
Conv, ReLU	$k, 1 \times 3$	$k, 3 \times 1$	n.a.
Dropout	$p_D$	$p_D$	n.a.
AvgPool	$F_T \times 1$	$1 \times T_K$	n.a.
Conv, ReLU	$64k, 1 \times T_T$	$64k, F_K \times 1$	n.a.
Dropout	$p_D$	$p_D$	n.a.

(b) DeepMod			
Layer	Temp	Spec	Square
Input			
Conv, ReLU	$2^\ell k, 1 \times 5$	$2^\ell k, 5 \times 1$	$2^\ell k, 5 \times 5$
BatchNorm			
Conv, ReLU	$2^\ell k, 1 \times 3$	$2^\ell k, 3 \times 1$	$2^\ell k, 3 \times 3$
BatchNorm			
MaxPool	$2 \times 2$	$2 \times 2$	$2 \times 2$
Dropout	$p_D$	$p_D$	$p_D$

(c) ClassMod			
Layer	Temp	Spec	Square
Input			
Conv, ReLU	$N_T, 1 \times 1$	$N_K, 1 \times 1$	n.a.
GlobalAvgPool			
Softmax			

Table 2: Layer definitions for the three modules ShallowMod, ClassMod, and DeepMod, describing number of filters (e.g.,  $k$  or  $64k$ ) and their respective shapes (e.g.,  $1 \times 3$  or  $5 \times 5$ ).

ShallowMod is followed by a fully convolutional classification module named ClassMod (Table 2c), which consists of a  $1 \times 1$  bottleneck layer (pointwise convolution) with as many filters as desired classes ( $N_K$  or  $N_T$ ), a global average pooling layer, and the softmax activation function. Note, that because all directional filters are identically aligned, the model has an asymmetric, *directional capacity*, i.e., it has a much larger ability to describe complex relationships in one direction than in the other.

We use the same general architecture for both key and tempo estimation. The only differences are the filter and pooling directions and dimensions. For tempo estimation we use temporal filters with pooling along the frequency axis, and for key estimation spectral filters with pooling along the time axis. Both architectures are named after their filter directions, ShallowTemp and ShallowSpec, respectively. We also adjust the pooling and the long filters shape to the size of the input spectrogram, which is different for the two tasks.

### 2.3.2 Deep Architectures

The second architecture, Deep (Table 1b), is a common VGG-style architecture consisting of six parameterized feature extraction modules DeepMod (Table 2b) followed by the same classification module that we have already used in Shallow. Each of the feature extraction modules contains a convolutional layer with  $5 \times 5$  filters followed by a convolutional layer with  $3 \times 3$  filters. The convolutional layers consist of  $2^\ell k$  filters each, with network parameter

Split	Key Datasets
Training	80% of LMD Key $\cup$ 80% of MTG Key
Validation	10% of LMD Key $\cup$ 20% of MTG Key
Testing	GiantSteps Key, GTzan Key, 10% of LMD Key

Split	Tempo Datasets
Training	80% of EBall $\cup$ 80% of MTG Tempo $\cup$ 80% of LMD Tempo
Validation	20% of EBall $\cup$ 20% of MTG Tempo $\cup$ 10% of LMD Tempo
Testing	GiantSteps Tempo, GTzan Tempo, 10% of LMD Tempo, Ballroom

Table 3: Dataset splits used in key (top) and tempo (bottom) estimation experiments.

$k$  and module parameter  $\ell$ . While  $\ell$  influences the number of filters in an instance of DeepMod,  $k$  lets us scale the total number of parameters in the network. As shown in Table 1b, deeper instances have more filters. The convolutional layers are followed by a  $2 \times 2$  max pooling layer. Should pooling not be possible along an axis, because the output from the previous layer is only 1 wide or high, pooling is skipped along that axis. This happens for example, when a tempo spectrogram with its 40 bands passes through more than 5 max pools. Each pooling layer is followed by a dropout layer with probability  $p_D$ . To counter covariate shift, we add batch normalization [22] layers after each convolutional layer.

The general structure of the Deep architecture is customized neither for the key nor for the tempo task. However, in order to investigate how different filter shapes affect the network’s performance, we modify the described architecture by replacing the square convolutional kernels with directional ones, i.e.,  $3 \times 3$  with  $1 \times 3$  or  $3 \times 1$ , and  $5 \times 5$  with  $1 \times 5$  or  $5 \times 1$ . Analogous to the naming scheme used for shallow networks, we denote the directional variants DeepTemp and DeepSpec. The original variant is named DeepSquare.

### 2.4 Datasets

We use the following publicly available datasets from different genres for both training and evaluation (listed in alphabetical order). The used splits are randomly chosen and listed in Table 3.

Ballroom (698 samples): 30 s excerpts with tempo annotations [23].

EBall (3,826 samples): 30 s excerpts with tempo annotations, excluding tracks also occurring in the regular Ballroom dataset [3, 23, 24].

GiantStepsKey (604 samples): 2 min excerpts of electronic dance music (EDM) [25].

GiantStepsTempo (661 samples): 2 min excerpts of EDM [25]. Revised tempo annotations from [26].



GTzan Key (836 samples): 30 s excerpts from 10 different genres [27]. Key annotations from [28].<sup>1</sup> Most tracks with missing key annotations belong to the genres classical, jazz, and hip-hop.

GTzan Tempo (999 samples): 30 s excerpts from 10 different genres [27]. Tempo annotations from [29].

LMD Key (6,981 samples): 30 s excerpts, predominantly rock and pop [30, 31]. Due to a MIDI peculiarity, this dataset does not contain any tracks in C major. Some form of data augmentation as described above is therefore necessary.

LMD Tempo (3,611 samples): 30 s excerpts, predominantly rock and pop [3, 30].

MTG Tempo / MTG Key (1,158 samples): 2 min EDM excerpts annotated with both key and tempo [3, 32]. We used only tracks that are still publicly available, have an unambiguous key, and a high key annotation confidence.<sup>2</sup>

## 2.5 Evaluation

Since the proposed network architectures are fully convolutional, we can choose at prediction time to pass a track either in one long spectrogram or as multiple shorter windows through the network. In the latter case, predictions for all windows would have to be aggregated. Informal experiments did not show a remarkable difference. For this work we choose to predict values for whole spectrograms.

When evaluating key estimation systems either a simple accuracy or a score is used that assigns additional value to musically justifiable mistakes, like being off by a perfect fifth.<sup>3</sup> For this work, we choose to only report the percentage of correctly classified keys. Tempo estimation systems are typically evaluated using the metrics *Accuracy1* and *Accuracy2*. While *Accuracy1* reports the percentage of correctly estimated tempi allowing a 4% tolerance, *Accuracy2* additionally permits so-called octave errors, i.e., errors by a factor of 2 and 3 [23]. We choose to report only *Accuracy1*.

For training we use Adam [33] as optimizer with a learning rate of 0.001, a batch size of 32, and early stopping once the validation loss has not decreased any more during the last 100 epochs. In this work, one epoch is defined as having shown all training samples to the network once, regardless of augmentation. We choose  $k$  so that we can compare architectures with similar parameter counts. Shallow is trained with  $k \in \{1, 2, 4, 6, 8, 12\}$  and Deep with  $k \in \{2, 4, 8, 16, 24\}$ . Additionally, DeepSquare is trained with  $k = 1$ . For both architectures we apply various dropout probabilities  $p_D \in \{0.1, 0.3, 0.5\}$ . Each variant is trained 5 times and mean validation accuracy along with its standard deviation is recorded for each variant. In total we train 420 models with 84 different configurations.

<sup>1</sup>[https://github.com/alexanderlerch/gtzan\\_key](https://github.com/alexanderlerch/gtzan_key)

<sup>2</sup><https://github.com/GiantSteps/giantsteps-mtg-key-dataset>

<sup>3</sup>[https://www.music-ir.org/mirex/wiki/2018:Audio\\_Key\\_Detection](https://www.music-ir.org/mirex/wiki/2018:Audio_Key_Detection)

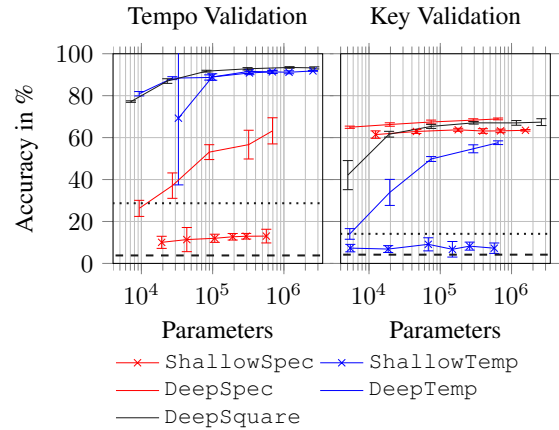


Figure 2: Mean validation accuracies for the various network configurations depending on their number of network parameters. Only the best dropout configuration is shown. Whiskers represent the standard deviation based on 5 runs.

For testing, we pick the dropout variant of each network class that performed best on the validation set and evaluate it against the test datasets. Again, we report the mean accuracies for 5 runs along with their standard deviations.

## 3. RESULTS

Figure 2 shows mean validation accuracies of 5 runs for each configuration, using their best performing dropout probability  $p_D$ . The dashed black line is the accuracy a random classifier achieves, and the dotted black line shows the accuracy of the algorithm that always outputs the class that most often occurs in the validation set. With accuracy values slightly above random, ShallowSpec and ShallowTemp perform worst of all architectures, when used for the task they were *not* meant for. But when used for the task they were designed for, both perform well. A higher number of parameters leads to slightly better results. When training ShallowTemp with  $k = 1$  for the tempo task, the network performed very poorly in one of the five runs, which is the cause for the very large standard deviation of 32.2 shown in Figure 2. The mean accuracy for the 4 successful runs was 85.2%. When comparing with the Deep architectures, we see that DeepTemp performs just as well as ShallowTemp with  $k > 1$  on the tempo task, and DeepSpec clearly outperforms ShallowSpec on the key task. Surprisingly, the DeepSpec architecture reaches fairly high accuracy values (up to 63%) on the tempo task when increasing the model capacity via  $k$ , even though it only has convolutional filters aligned with the frequency axis. We can make a similar observation for the DeepTemp architecture. It too reaches relatively high accuracy values on the key task (up to 57%) when increasing  $k$ . The unspecialized DeepSquare is by a small margin the best performing architecture for the tempo task, and comes in as a close second for key detection with  $k > 1$ . But for  $k = 1$ , DeepSquare performs considerably worse than DeepSpec with  $k = 2$  (42% compared to 64%), even though both have similar parameter counts



of ca. 5 000.

We selected the dropout variant for each architecture and parameter setting with the best validation accuracy and ran predictions on the test sets. Detailed results are shown in Figure 3. The general picture is very similar to validation: Deep architectures tend to perform slightly better than Shallow architectures on the tasks they are meant for and Shallow architectures perform poorly on the task they were not meant for. In fact, ShallowTemp performs no better on GTzanKey and GiantStepsKey than the random baseline. For both key and tempo DeepSquare performs as well or better than any other architecture, except when drastically reducing the model capacity for the key task ( $k = 1$ ). Then accuracy decreases well below DeepSpec's performance with similar parameter count: 33% compared to 50% for GTzanKey, and 21% compared to 51% for GiantStepsKey.

To provide an absolute comparison, we chose the best performing representative from each architecture (based on validation accuracy, regardless of dropout configuration or capacity), and calculated accuracies for each test set (Table 4, incl. reference values from the literature). In 5 out of 7 test cases DeepSquare reaches the highest accuracy score among our architectures. The other two are reached by DeepTemp for GiantStepsTempo and by DeepSpec for LMD Key. For both tasks we observe that the margin by which the best performing network is better than the second best for a given dataset differs considerably. DeepSquare reaches an accuracy of 92.4% for the Ballroom tempo dataset, which is 4.2 pp (percentage points) better than the second best network (DeepTemp, 88.2%). The differences between best and second best accuracy are considerably lower for the other datasets: 1.7 pp (LMD Tempo), 1.6 pp (GTzan Tempo), and 0.6 pp (GiantSteps Tempo). For the key task, DeepSquare reaches an accuracy of 49.9% on GTzanKey, which is 5.1 pp better than the second best network (DeepSpec, 44.8%), while the differences between best and second best for the other datasets are 3.1 pp (GiantSteps Key), and 2.4 pp (LMD Key).

#### 4. DISCUSSION

The results show that simple shallow networks with axis-aligned, directional filters can perform well on both the key and tempo task. Conceptually, both tasks are similar enough that virtually the same architecture can be used for either one, as long as the input representation and the filter direction are appropriate. Using the wrong filter direction, e.g., ShallowSpec for the tempo task, leads to very poor results close to the random baseline. Together, this strongly supports the hypothesis that the Shallow architecture indeed learns what we want it to learn, i.e., pitch patterns for key detection or rhythmic patterns for tempo detection, but not both.

This stands in contrast to the standard VGG-style network (DeepSquare). Because of its square filters, we cannot be certain what it learns, just by analyzing its static architecture. It is designed to pick up on anything that could provide a hint towards correct classification, be it

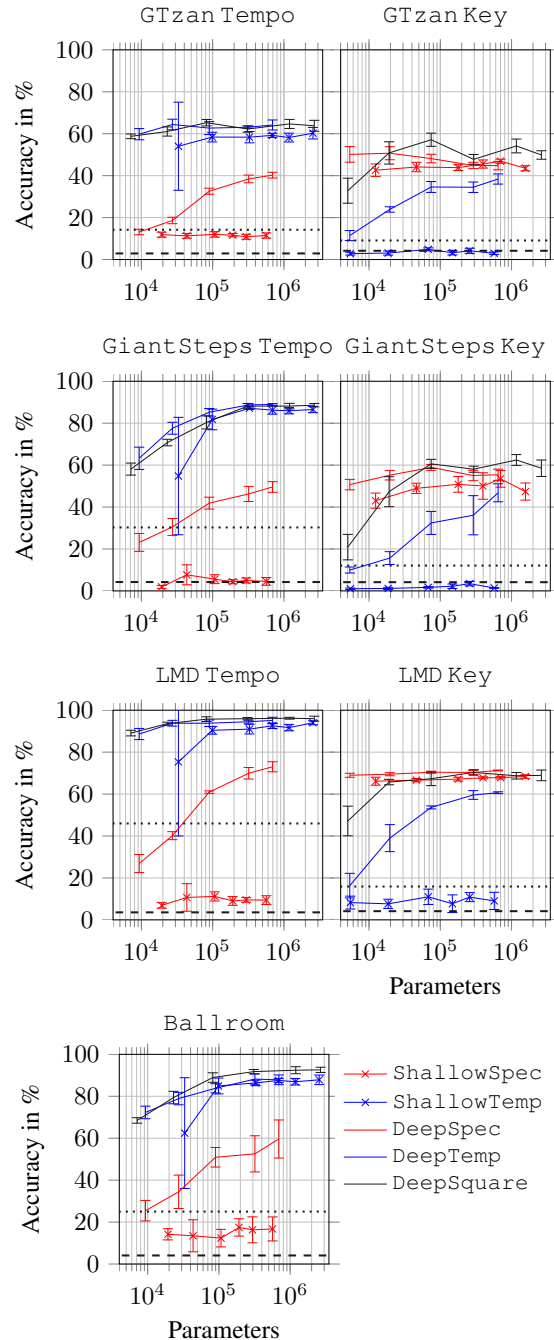


Figure 3: Mean test accuracies for various network configurations and datasets depending on their number of network parameters. Whiskers represent one standard deviation based on 5 runs. Dropout was chosen based on performance during validation.

rhythm and pitch patterns, or timbral properties like instrumentation. And indeed our experiment shows that without being specialized for either key or tempo estimation in any way, DeepSquare works very well for both tasks. In Section 3 we noted that DeepSquare achieved the greatest tempo accuracy for Ballroom and the greatest key accuracy for GTzan Key by a considerable margin of 4.2 pp

Architecture	GS	GT	LMD	BR	GS	GT	LMD
ShallowTemp	86.5 <sup>1.5</sup>	60.3 <sup>2.7</sup>	94.0 <sup>1.0</sup>	87.9 <sup>2.3</sup>	1.7 <sup>0.4</sup>	4.9 <sup>0.7</sup>	11.0 <sup>3.7</sup>
DeepTemp	<b>88.7</b> <sup>0.6</sup>	63.1 <sup>0.6</sup>	94.5 <sup>0.7</sup>	88.2 <sup>2.4</sup>	46.8 <sup>4.3</sup>	38.4 <sup>2.4</sup>	60.7 <sup>0.4</sup>
ShallowSpec	4.5 <sup>1.9</sup>	11.5 <sup>1.3</sup>	9.4 <sup>2.1</sup>	16.7 <sup>5.7</sup>	50.8 <sup>3.8</sup>	43.8 <sup>1.4</sup>	67.1 <sup>0.9</sup>
DeepSpec	49.6 <sup>2.5</sup>	40.2 <sup>1.4</sup>	73.0 <sup>2.4</sup>	59.6 <sup>9.1</sup>	55.4 <sup>2.7</sup>	44.8 <sup>2.0</sup>	<b>71.3</b> <sup>0.2</sup>
DeepSquare	88.1 <sup>1.3</sup>	<b>64.7</b> <sup>2.1</sup>	<b>96.2</b> <sup>0.4</sup>	<b>92.4</b> <sup>1.7</sup>	<b>58.5</b> <sup>3.9</sup>	<b>49.9</b> <sup>2.0</sup>	68.9 <sup>2.5</sup>
Literature	82.5 [3]	78.3 [29]	—	92.0 [3]	67.9 [2]	~45 [28]	—

(a) Tempo

(b) Key

Table 4: Mean estimation accuracies of 5 runs with standard deviation (small font). Best results per test are set in **bold**. Model variants chosen based on validation performance (ignoring parameter count). GS=GiantSteps, GT=GTzan, LMD=LMD, BR=Ballroom.

and 5.1 pp, respectively. This margin may be a result of the fact that key and tempo are related to genre [34–37]. Specifically, in Ballroom there is a strong correlation between genre and tempo, and GTzanKey is the key test set with the greatest genre diversity and therefore stands to benefit the most from an architecture that can distinguish genres based on *both* temporal and timbral properties. Consequently, square filters *should* improve accuracy results for these datasets. But this does not conclusively show that only the network’s ability to measure specifically key or tempo is reflected by these results, as the system is by design vulnerable to confounds [15]. By using directional filters in DeepSpec and DeepTemp we intentionally limit the standard VGG-style architecture in a way that seeks to lessen this vulnerability as well as reduce the number of required parameters.

The results for DeepSpec and DeepTemp show that a VGG-style network with directional filters can perform very well on either task. For networks with a large number of parameters test results are similar to DeepSquare, with a tendency towards a slightly worse performance. Interestingly, the situation is different for low-capacity networks with  $k = 2$  for DeepSpec, and  $k = 1$  for DeepSquare. Here, DeepSpec clearly outperforms DeepSquare, even though the parameter count is similar. Perhaps with ca. 5 000 parameters DeepSquare simply does not have enough capacity aligned in the right direction to still perform well on the task.

The fact that DeepSpec and DeepTemp with  $k = 2$  perform very poorly on the tasks they are not meant for, supports the hypothesis that they only learn the intended features for the tasks they are meant for. For  $k > 2$  we cannot be quite as certain, as both architectures reach higher accuracy scores on the tasks they were not meant for for greater values of  $k$ . We believe this effect may be a result of the  $2 \times 2$  max pooling in the DeepMod modules.

## 5. CONCLUSIONS

We have shown that shallow, signal processing-inspired CNN architectures using directional filters can be used successfully for both tempo and key detection. By using shallow networks designed for key detection on the tempo task and vice versa, we were able to experimentally support

the hypothesis that these networks are incapable of matching information from the domain they were not meant for, which would make them less susceptible to confounds.

We further demonstrated that a standard VGG-style architecture can be used for tempo estimation, as it has been shown before for key detection [2]. By replacing square filters with directional filters, we derived a musically motivated, directional VGG-variant that performs similarly well as the original one, but is less vulnerable to confounds, especially when used for key detection with low capacity models. In such scenarios we were also able to observe efficiency gains, i.e., better performance than the standard VGG-style network with similar parameter counts.

## Additional Material

Code to recreate models and reproduce the reported results can be found at [https://github.com/hendriks73/directional\\_cnns](https://github.com/hendriks73/directional_cnns).

## Acknowledgments

The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits IIS. Meinard Müller is supported by the German Research Foundation (DFG MU 2686/11-1).

## 6. REFERENCES

- [1] F. Korzeniowski and G. Widmer, “End-to-end musical key estimation using a convolutional neural network,” in *Proceedings of the 25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece, 2017.
- [2] —, “Genre-agnostic key classification with convolutional neural networks,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, September 2018.
- [3] H. Schreiber and M. Müller, “A single-step approach to musical tempo estimation using a convolutional neural network,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, September 2018.

- [4] A. Holzapfel and T. Grill, “Bayesian meter tracking on learned signal representations,” in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, New York, NY, USA, August 2016, pp. 262–268.
- [5] S. Durand, J. P. Bello, B. David, G. Richard, S. Durand, J. P. Bello, B. David, G. Richard, B. David, G. Richard *et al.*, “Robust downbeat tracking using an ensemble of convolutional networks,” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 1, pp. 76–89, 2017.
- [6] A. Gkiokas and V. Katsouros, “Convolutional neural networks for real-time beat tracking: A dancing robot application,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, Suzhou, China, October 2017, pp. 286–293.
- [7] S. Dieleman, P. Brakel, and B. Schrauwen, “Audio-based music classification with a pretrained convolutional network,” in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011, pp. 669–674.
- [8] A. Schindler, T. Lidy, and A. Rauber, “Comparing shallow versus deep neural network architectures for automatic music genre classification,” in *Proceedings of the 9th Forum Media Technology (FMT2016)*, vol. 1734, St. Pölten, Austria, November 2016, pp. 17–21.
- [9] M. Dorfer and G. Widmer, “Training general-purpose audio tagging networks with noisy labels and iterative self-verification,” DCASE2018 Challenge, Tech. Rep., September 2018.
- [10] K. Choi, G. Fazekas, and M. B. Sandler, “Automatic tagging using deep convolutional neural networks,” in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, New York, NY, USA, August 2016, pp. 805–811.
- [11] B. McFee and J. P. Bello, “Structured training for large-vocabulary chord recognition,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, Suzhou, China, October 2017, pp. 188–194.
- [12] J. Pons, T. Lidy, and X. Serra, “Experimenting with musically motivated convolutional neural networks,” in *Proceedings of 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, Bucharest, Romania: IEEE, June 2016, pp. 1–6.
- [13] F. Gouyon, S. Dixon, E. Pampalk, and G. Widmer, “Evaluating rhythmic descriptors for musical genre classification,” in *AES 25th International Conference*, London, UK, 2004.
- [14] J. Pons and X. Serra, “Designing efficient architectures for modeling temporal features with convolutional neural networks,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. New Orleans, USA: IEEE, March 2017, pp. 2472–2476.
- [15] B. L. Sturm, “A simple method to determine if a music information retrieval system is a “horse”,” *IEEE Transactions on Multimedia*, vol. 16, no. 6, pp. 1636–1644, October 2014.
- [16] —, “Classification accuracy is not enough,” *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 371–406, 2013.
- [17] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich *et al.*, “Going deeper with convolutions,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, June 2015, pp. 1–9.
- [19] A. P. Klapuri, “Sound onset detection by applying psychoacoustic knowledge,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Washington, DC, USA, 1999, pp. 3089–3092.
- [20] C. L. Krumhansl, *Cognitive foundations of musical pitch*, ser. Oxford Psychology Series. Oxford University Press, 1990, no. 17.
- [21] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [22] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.
- [23] F. Gouyon, A. P. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, “An experimental comparison of audio tempo induction algorithms,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1832–1844, 2006.
- [24] U. Marchand and G. Peeters, “The extended ballroom dataset,” in *Late Breaking Demo of the International Conference on Music Information Retrieval (ISMIR)*, New York, NY, USA, 2016.
- [25] P. Knees, Á. Faraldo, P. Herrera, R. Vogl, S. Böck, F. Hörschläger, and M. Le Goff, “Two data sets for tempo estimation and key detection in electronic dance music annotated from user corrections,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, October 2015, pp. 364–370.
- [26] H. Schreiber and M. Müller, “A crowdsourced experiment for tempo estimation of electronic dance music,” in *Proceedings of the 19th International Society*

for *Music Information Retrieval Conference (ISMIR)*, Paris, France, September 2018.

- [27] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.
- [28] S. Kraft, A. Lerch, and U. Zölzer, “The tonalness spectrum: feature-based estimation of tonal components,” in *Proceedings of the 16th International Conference on Digital Audio Effects (DAFx)*, Maynooth, Ireland, September 2013, p. 8.
- [29] G. Percival and G. Tzanetakis, “Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses,” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 12, pp. 1765–1776, 2014.
- [30] C. Raffel, “Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching,” Ph.D. dissertation, Columbia University, 2016.
- [31] H. Schreiber, “CNN-based automatic musical key detection,” in *Music Information Retrieval Evaluation eXchange (MIREX)*, Suzhou, China, October 2017.
- [32] Á. Faraldo, S. Jordà, and P. Herrera, “A multi-profile method for key estimation in EDM,” in *Proceedings of the AES International Conference on Semantic Audio*. Erlangen, Germany: Audio Engineering Society, June 2017.
- [33] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [34] F. Hörschläger, R. Vogl, S. Böck, and P. Knees, “Addressing tempo estimation octave errors in electronic music by incorporating style information extracted from wikipedia,” in *Proceedings of the Sound and Music Computing Conference (SMC)*, Maynooth, Ireland, 2015.
- [35] B. Schuller, F. Eyben, and G. Rigoll, “Tango or waltz?: Putting ballroom dance style into tempo detection,” *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2008, p. 12, 2008.
- [36] L. Xiao, A. Tian, W. Li, and J. Zhou, “Using statistic model to capture the association between timbre and perceived tempo,” in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, USA, 2008.
- [37] Á. Faraldo, E. Gómez, S. Jordà, and P. Herrera, “Key estimation in electronic dance music,” in *European Conference on Information Retrieval*. Springer, 2016, pp. 335–347.



# THE VIKING HRTF DATASET

**Simone Spagnol**

Dept. of Architecture, Design & Media Technology  
Aalborg University  
Copenhagen, Denmark  
ssp@create.aau.dk

**Kristján Bjarki Purkhús, Rúnar Unnthórsson**

School of Engineering and Natural Sciences  
University of Iceland  
Reykjavík, Iceland  
runson@hi.is

**Sverrir Karl Björnsson**

Dept. of Mechanical Engineering  
Technical University of Denmark  
Kgs. Lyngby, Denmark  
s171951@student.dtu.dk

## ABSTRACT

This paper describes the Viking HRTF dataset, a collection of head-related transfer functions (HRTFs) measured at the University of Iceland. The dataset includes full-sphere HRTFs measured on a dense spatial grid (1513 positions) with a KEMAR mannequin with 20 different artificial left pinnae attached, one at a time. The artificial pinnae were previously obtained through a custom molding procedure from 20 different lifelike human heads. The analyses of results reported here suggest that the collected acoustical measurements are robust, reproducible, and faithful to reference KEMAR HRTFs, and that material hardness has a negligible impact on the measurements compared to pinna shape. The purpose of the present collection, which is available for free download, is to provide accurate input data for future investigations on the relation between HRTFs and anthropometric data through machine learning techniques or other state-of-the-art methodologies.

## 1. INTRODUCTION

Binaural sound rendering techniques typically rely on the use of Head-Related Transfer Functions (HRTFs), i.e. filters that capture the acoustic effects of the human head [1]. HRTFs allow accurate simulation of the signal that arrives at the entrance of the ear canal as a function of the spatial location of the sound source (azimuth, elevation, and distance). The ideal rendering in terms of accuracy involves the use of individual HRTFs measured on the listener. By processing a desired monophonic sound signal with a pair of individual HRTFs, one per channel, and by adequately accounting for headphone-induced spectral coloration [2], authentic 3D sound experiences can take place. Virtual sound sources created with individual HRTFs can be localized almost as accurately as real sources and externalized,

provided that head movements can be made [3]. Binaural technologies have a wide variety of applications, ranging from personal entertainment [4], through immersive virtual environments [5], to travel aids for the blind [6].

However, obtaining individual HRTF data is only possible with dedicated research facilities and invasive and/or strenuous recording procedures [7]. This is the reason why non-individual HRTFs, acoustically measured on anthropomorphic mannequins or generic human individuals, are often preferred in practice. Several HRTF sets are available online, the most popular being those measured on the KEMAR mannequin [8] or the Neumann KU-100 dummy head [9]. Alternatively, an HRTF set can be taken from one of many public databases of individual measurements (e.g. the CIPIC database [10]); many of these databases were recently unified in a common HRTF format known as Spatially Oriented Format for Acoustics (SOFA).<sup>1</sup>

The drawback with non-individual HRTFs is that they obviously refer to a different anthropometry than the listener's. In particular, the most relevant differences between the HRTFs of two subjects are due to different pinna features (shape, size, and orientation) that give every individual a unique pinna shape [11]. When used for binaural rendering, this HRTF mismatch often results in localization errors such as front/back confusion, wrong perception of elevation, and inside-the-head localization [12].

For the above reasons, studying the relationship between pinna features and HRTFs is an essential step towards understanding of the underlying acoustical mechanisms. It is known, for instance, that the frequency of the first pinna-related peak depends on the dimensions of the concha [13, 14], and that notch frequencies are related to the distance between the ear canal and the most prominent pinna edges [15, 16]. However, previous studies that include applications of anthropometric regression methods to measured HRTF data [17–19] have produced mixed results, highlighting that many of these relations are not fully understood yet. One of the reasons might rely in the HRTF data collected on a human population, with issues related to the relative positioning of the microphones inside the

Copyright: © 2019 Simone Spagnol et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> [www.sofaconventions.org](http://www.sofaconventions.org)





Figure 1. The used HRTF measurement system and room (left); detail of the KEMAR with custom left pinna (right).

ears, head movements during measurement, and non-ideal measurement conditions. Still, even measurements on the same dummy head often result in dissimilar HRTF sets, depending on the used measurement system [9].

In this paper we present the design, implementation and preliminary analysis of a novel set of HRTF measurements, that we refer to as the Viking HRTF dataset. Measurements were taken on a KEMAR mannequin with various interchangeable pinna shapes, with the purpose of providing accurate input data for determining the individual HRTF from a 3D representation of the user [20]. We implemented a procedure for casting artificial pinnae from lifelike human heads so as to provide a reasonable sample of pinna shapes for KEMAR from a real human population, having all remaining anthropometric parameters of the head, torso, and opposite pinna fixed. Furthermore, although limited by the available resources, we designed the automated measurement setup to be as accurate and stable as possible, in order to guarantee reproducible measurements.

## 2. METHODS

### 2.1 Measurement system

The HRTF measurements were taken automatically with the system pictured in Figure 1. The measurement system consisted of a KEMAR mannequin<sup>2</sup> mounted on a 360° rotating cylindrical stand and a Genelec 8020CPM-6 loudspeaker mounted on an L-shaped rotating arm. The configuration for the mannequin was the 45BB-4 with standard large anthropometric pinnae (35 Shore-OO hardness, GRAS KB5000/01) as reference and half-inch pressure microphones (GRAS 40AO) placed at the ear canal entrances. The two audio channels were fed to an RME Fireface 802 sound card connected to a PC running MATLAB.

The distance between the loudspeaker tweeter and center of the mannequin head was designed to be a constant

<sup>2</sup> <http://kemar.us>



Figure 2. The making of a negative ear mold: (a) 1st negative mold; (b) Jesmonite® replica; (c) 2nd negative mold.

1 m, independently of the orientation of the mannequin and arm. This distance value guarantees the collection of far-field spectral cues with reasonable accuracy [21]. Absolute references for azimuth and elevation were provided through a fixed marker on the bottom of the mannequin (0° azimuth) and a bubble level mounted on the side segment of the arm (0° elevation), respectively. Correct alignment in all three axes could be attained by targeting laser beams to the microphones and tip of the nose of the KEMAR. This experimental choice allows collection of HRTFs on a full-azimuth range with elevations from  $-50^\circ$  to  $90^\circ$ , according to a vertical polar coordinate system.

Rotation of the mannequin and arm was managed by two independent high-torque step motors (JVL MST001A, 1.2 Nm), controlled through two digital step drives (Geckodrives G213V) in full-step mode (200 steps per revolution). In order to increase the torque and angular resolution of the system, two 100 : 1 gearboxes were installed between the first motor and the arm and between the second motor and the mannequin. As a result, the minimum rotation angle of both the arm and mannequin was  $0.018^\circ$ . Furthermore, in order to reduce the needed torque on the arm, a 22 kg counterweight was applied to the shorter appendix so to balance the weight of the loudspeaker and the longer appendix of the arm. Communication between the step drives and PC was also managed in MATLAB.

### 2.2 Pinna casting

In order to provide a sufficient sample of pinna shapes for the HRTF measurements, we set up and applied a custom

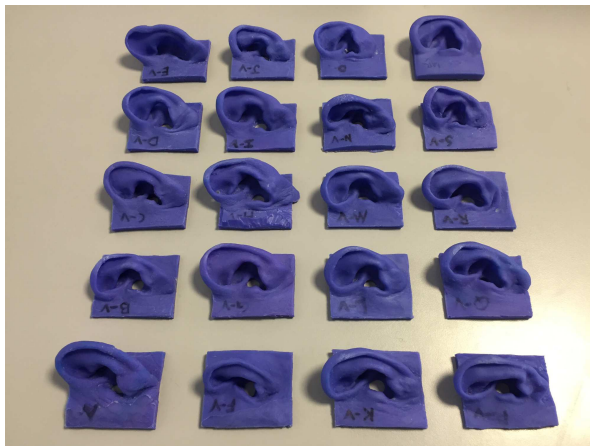


Figure 3. The 20 left pinna replicas obtained through the casting procedure.

procedure to cast silicone replicas of pinnae from dummy heads to fit the KEMAR as follows.

*Step 1: First negative mold.* We applied two layers of silicone mix to the ear, first a thin one with a paintbrush and then a thick one (1/2 cup) with a spatula, letting each layer dry for 1 day before the next step. In order to reduce leakage onto vertical surfaces, we added a few drops of thixotropic additive to the silicone for the thick layer. Then, we applied wet plaster of Paris strips over the negative silicone mold (Figure 2a) in order to efficiently peel it from the dummy head and to give it support so it does not deform in Step 2.

*Step 2: Jesmonite® replica.* We poured a quick mix of 1/2 cup of water and two teaspoons of Jesmonite® into the negative mold obtained in Step 1 placed above a vibration table (in order to avoid formation of air bubbles), and let it harden for 1 – 2 days. Then, we removed the Jesmonite® ear (Figure 2b) and occasionally filled gaps or deformations in it with clay. Finally, we drilled a hole in the back of the concha and sanded the ear base with a belt sander to accurately fit the size of the KEMAR pinna slot.

*Step 3: Second negative mold.* We placed the Jesmonite® ear replica into a custom-made plastic box with open top and secured it to the bottom with clay. Then, we poured 2 cups of silicone mix inside the plastic box, placed on the vibration table, and let it dry for 1 day. We finally removed the plastic box and Jesmonite® ear (Figure 2c).

*Step 4: Final pinna replica.* We first poured car wax into the inner parts of the negative mold obtained in Step 3 as a release agent, and let it dry for half a day. Then we poured 1/3 cup of a different silicone mix from the one used in Steps 1 and 3 (25 Shore-A hardness) into the negative mold, and let it dry for 1 day. We finally removed the silicone pinna replica and cut the excess parts on the base with a knife.

As pictured in Figure 3, we applied the procedure to a series of left ears of 20 different subjects. These include the KEMAR with standard large anthropometric pinnae and 19 different lifelike dummy heads made out of plaster, borrowed from the Saga Museum in Reykjavík. The

Table 1. HRTF measurement positions.

Elevations [deg]	[-45,45]	[50,70]	[75,85]	90
Step [deg]	5	15	45	360
No. of azimuths	72	24	8	1

heads, manufactured between 2001 and 2003 for artistic purposes,<sup>3</sup> reproduce with high fidelity the anthropometric traits of 19 Icelandic humans (7 female), aged between 7 and 77 at the time of manufacturing.

### 2.3 Measurement procedure

The measurements took place on several days outside of office hours (4PM to 12PM) in a silent office room with reflecting walls, floor and ceiling inside the Tæknigarður building of the University of Iceland (see left panel of Figure 1). Since we did not record the free-field response, the measurements include the response of the loudspeaker and microphones as well as some effects of the room (later minimized by windowing as explained in the next subsection). We used the logarithmic sweep method [22] to record the single acoustical responses. The input sweep signal, whose level was kept constant throughout the whole measurement schedule, spanned frequencies between 20 Hz and 20 kHz in 1 s, at a sampling rate  $f_s = 48$  kHz. The average SPL level at 1 kHz for a frontal stimulus as collected through a Class 1 sound level meter at the center of the reference system was 82 dB.

Sound source location was specified through the azimuth angle  $\theta$  and elevation angle  $\phi$  in vertical-polar coordinates. Elevations were uniformly sampled in  $5^\circ$  steps from  $-45^\circ$  to  $90^\circ$ , while azimuths were sampled in different increments as shown in Table 1 in order to obtain roughly uniform density towards the upper pole of the sphere. The total number of spatial positions per measurement was 1513. At each measurement session, and for each elevation, the mannequin was consecutively rotated of the corresponding angular step and sweep responses were acquired at each azimuth angle. After completion of all azimuth angles for the current elevation, the mannequin was rotated back to its starting position and the arm moved down to the next elevation angle. In order to maintain the HRTF measurements as silent as possible, 0.5-ms pauses were introduced between all motor commands and the previous/next recording. The duration of one single measurement session was approximately 105 minutes.

Twenty-three measurement sessions (3 control measurements + 20 test measurements) were scheduled in total. A standard large anthropometric pinna (35 Shore-OO hardness) was installed on the right channel of the KEMAR throughout the whole measurement schedule, while the left pinna changed at every measurement session. In the first and second control sessions (*CS1* and *CS2*) we installed the corresponding original left KEMAR pinna and its variant with different hardness (55 Shore-OO), respectively. These sessions were introduced in order to check for dif-

<sup>3</sup> <https://sagamuseum.is/sagadesign/>

ferences in materials and to assess the fidelity of our KEMAR pinna replica to the original ones. The following 20 sessions were devoted to the 20 custom-made pinnae, including the KEMAR pinna replica (right panel of Figure 1) and the 19 human pinnae (alphabetically labeled *A* to *S*, see Figure 3). The third and last control session (*CS3*) was again a measurement with both original 35 Shore-OO hardness pinnae, and was introduced in order to control for reproducibility of our measurements. Between each two measurement sessions, the HRTF measurement system was manually calibrated to the correct starting position by tuning each motor to its corresponding absolute reference.

## 2.4 Post-processing

A post-processing script was written to recover the HRTF from the corresponding raw sweep response, based on the following sequential steps. First, we compute the cross-correlation function  $\Psi[n]$  between the raw sweep response  $y[m]$  and the input sweep signal  $s[m]$  in order to find the starting point of the sweep response. In particular, we find the maximum correlation value  $\Psi_M$  in  $\Psi[n]$  and extract the lag  $n_i$  of the first sample in  $\Psi[n]$  such that  $\Psi[n] > 0.5\Psi_M$ . Then, the starting point of  $y[m]$  corresponds to the lag  $n_j$  with maximum correlation value within a 21-sample interval centered in  $\Psi(n_i)$ . This trick was introduced so as to avoid directly picking  $\Psi_M$  as starting point, that due to the echoic measurement conditions could correspond to a wall reflection, especially in the case of contralateral HRTFs.

Then, according to the logarithmic sweep method, we perform inverse filtering on the measured sweeps in order to obtain the corresponding impulse responses. The inverse reference spectrum  $S^{-1}[n]$  of the excitation signal is first computed and then low-passed and high-passed with second-order digital Butterworth filters to compensate for the original zero sound pressure level below 20 Hz and above 20 kHz in the sweep signal. Then, the impulse response is obtained as

$$h[n] = \Re(\mathcal{F}^{-1}(\mathcal{F}(y[n]) * S^{-1}[n])), \quad (1)$$

where  $\mathcal{F}$  and  $\mathcal{F}^{-1}$  are the DFT and inverse DFT functions, respectively.

Subsequently, a 128-sample Hann window is applied to each impulse response  $h[n]$  with the aim of removing unwanted early and late reflections occurring later than approximately 2.5 ms from the onset on the measurement equipment and inside the room, yielding  $h_w[n]$ . This value guarantees an appropriate windowing even for those measurement points where the loudspeaker is close to the floor or the ceiling. The HRTF is then simply calculated as the magnitude response of the DFT of  $h_w[n]$ .

## 3. RESULTS

In this section we show some preliminary results as an assessment of the trustworthiness of the collected HRTF dataset. Figure 4 shows an example plot of HRTF magnitudes on the median plane for a custom-made pinna (set *J*). Well-known effects can be recognized here, such as

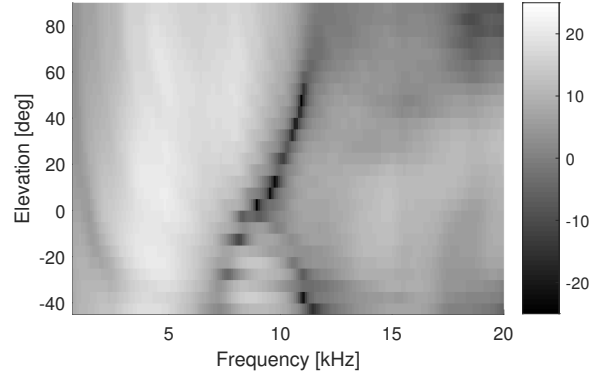


Figure 4. Median-plane HRTF magnitudes of set *J*, left channel.

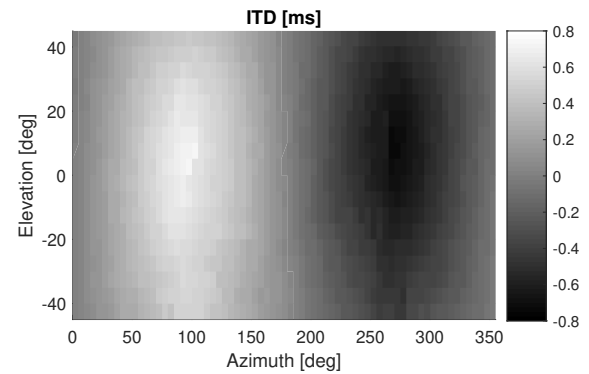


Figure 5. ITD measurements of set *J* (left channel: custom made pinna, right channel: KEMAR 35 Shore-OO pinna).

the shoulder reflection ridge between 1 and 2 kHz, the omnidirectional resonance around 4 kHz, and the elevation-dependent pattern of peaks and notches [1], with the higher number of notches at lower elevations [23, 24].

We also computed interaural time differences (ITD) as the offset between the starting points of the left and right channel for each sweep response on all sets. Figure 5 shows ITDs for the same example set. In accordance with previous literature [25], ITD values range between zero (median plane) and  $\pm 0.8$  ms for lateral directions. We also report in general noticeable asymmetries between left and right channels, resulting in ITD maxima and minima at two different elevation angles. This is due to the presence of two different pinnae in most measurements (KEMAR on the right channel and custom-made on the left one).

In order to evaluate the robustness and reproducibility of our measurements, we calculated the mean spectral distortion between each pair of HRTF sets  $H_a$  and  $H_b$  as [26]

$$SD(a, b) = \frac{1}{N_\alpha} \sum_i \sqrt{\frac{1}{N_f} \sum_k \left( 20 \log_{10} \frac{|H_a(\alpha_i, f_k)|}{|H_b(\alpha_i, f_k)|} \right)^2}, \quad (2)$$

where  $\alpha_i$  is an available spatial position,  $f_k$  is an available frequency,  $N_\alpha$  is the number of common spatial positions, and  $N_i$  is the number of frequencies in the con-



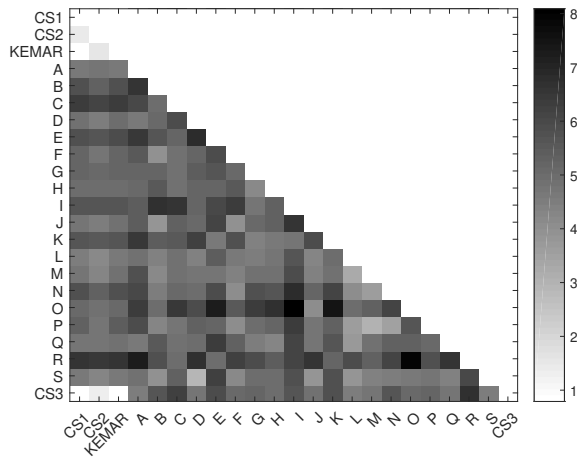


Figure 6. Mean spectral distortion [dB] between all pairs of HRTF sets (left channels only). For simplicity, values above the diagonal line are not repeated.

sidered range. For the following computations, in order to best capture the effect of the pinna, we limit this range between 3 kHz and 10 kHz. While the right channel is just affected by measurement noise, as the low mean spectral distortion (mean 0.26 dB, std 0.07 dB) demonstrates, the left channel results show considerable differences between different pinnae. As displayed in Figure 6, mean spectral distortion between each pair of custom-made human pinnae is never less than 2.93 dB (mean 5.25 dB, std 0.87 dB).

On the other hand, the KEMAR pinna replica scores a lower mean distortion with respect to the reference set measured in *CS1* (0.84 dB), which is interestingly lower than that of the original KEMAR 55 Shore-OO pinna measured in *CS2* (1.37 dB). This result suggests the accuracy of the pinna casting procedure and the negligible impact of the silicone hardness. In other words, although differences in hardness may slightly influence the spectral similarity of HRTFs, even the most minimal difference in pinna shape seems more influential. Furthermore, results of *CS3* indicate that although repositioning the KEMAR pinna on the left channel introduces some additional distortion (0.79 dB), this does not exceed twice the distortion due to measurement noise (right channel, 0.42 dB).

Finally, as an assessment of the fidelity of our KEMAR measurements, in Figure 7 we show the current measurements of the KEMAR with the reference pinna (35 Shore-OO hardness, measured in control session 1) on the horizontal plane and compare them to previous high-quality measurements of the KEMAR mannequin at 1-m distance by Brungart and Rabinowitz (top right panel of Figure 5 in [27]). If we exclude the low-frequency region, where differences are clearer because of the different ear-canal configuration of the KEMAR in the two measurements, we can recognize the same salient spectral features above 2 kHz, corresponding to the five reference points A-E (spectral peaks and notches). This qualitative result evidences the effectiveness of our measurement setup in conveying high-quality reference KEMAR measurements.

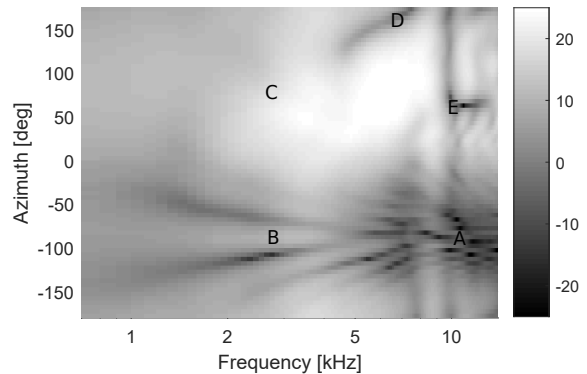


Figure 7. Reference horizontal-plane KEMAR measurements, left channel.

#### 4. CONCLUSIONS

In this paper we presented a dataset of full-sphere HRTFs of KEMAR with 20 different left pinnae obtained from molds of lifelike human heads. The measurement setup and procedure was described in detail, along with the post-processing operations designed to extract polished HRTFs from measured responses. Our preliminary results suggest the accuracy, variety, and reproducibility of the collected data. On the other hand, the hardness finding warrants quantified replications with a measure of central tendency and dispersion. These results will hopefully encourage investigations on the relation between HRTFs and anthropometric data through machine learning techniques or other state-of-the-art methodologies [28]. Full HRTF and ITD data, together with detailed scans of the used pinnae, are available for free download at the dataset web page<sup>4</sup> that will constantly be updated with new data and information.

#### Acknowledgments

The authors would like to thank Ernst Backman, owner of the Saga Museum in Reykjavík, for his invaluable assistance with the ear casting process and the access to his workshop and tools. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreements No 643636 and No 797850.

#### 5. REFERENCES

- [1] C. I. Cheng and G. H. Wakefield, "Introduction to head-related transfer functions (HRTFs): Representations of HRTFs in time, frequency, and space," *J. Audio Eng. Soc.*, vol. 49, no. 4, pp. 231–249, April 2001.
- [2] Z. Schärer and A. Lindau, "Evaluation of equalization methods for binaural signals," in *Proc. 126th Conv. Audio Eng. Soc.*, Munich, Germany, May 2009.
- [3] A. W. Bronkhorst, "Localization of real and virtual sound sources," *J. Acoust. Soc. Am.*, vol. 98, no. 5, pp. 2542–2553, November 1995.

<sup>4</sup> [itsadive.create.aau.dk/index.php/viking-hrtf/](https://itsadive.create.aau.dk/index.php/viking-hrtf/)

- [4] D. S. Brungart, "Near-field virtual audio displays," *Presence*, vol. 11, no. 1, pp. 93–106, February 2002.
- [5] S. Spagnol, R. Hoffmann, F. Avanzini, and A. Kristjánsson, "Effects of stimulus order on auditory distance discrimination of virtual nearby sound sources," *J. Acoust. Soc. Am.*, vol. 141, no. 4, pp. EL375–EL380, April 2017.
- [6] S. Spagnol, G. Wersényi, M. Bujacz, O. Balan, M. Herrera Martínez, A. Moldoveanu, and R. Unnthórsson, "Current use and future perspectives of spatial audio technologies in electronic travel aids," *Wireless Comm. Mob. Comput.*, vol. 2018, p. 17 pp., March 2018.
- [7] B. Xie, *Head-Related Transfer Function and Virtual Auditory Display*, 2nd ed. Plantation, FL, USA: J.Ross Publishing, June 2013.
- [8] M. D. Burkhard and R. M. Sachs, "Anthropometric manikin for acoustic research," *J. Acoust. Soc. Am.*, vol. 58, no. 1, pp. 214–222, July 1975.
- [9] A. Andreopoulou, D. R. Begault, and B. F. G. Katz, "Inter-laboratory round robin HRTF measurement comparison," *IEEE J. Select. Topics Signal Process.*, vol. 9, no. 5, pp. 895–906, August 2015.
- [10] V. R. Algazi, R. O. Duda, D. M. Thompson, and C. Avendano, "The CIPIC HRTF database," in *Proc. IEEE Work. Appl. Signal Process., Audio, Acoust.*, New Paltz, New York, USA, October 2001, pp. 1–4.
- [11] A. Abaza, A. Ross, C. Hebert, M. A. F. Harrison, and M. S. Nixon, "A survey on ear biometrics," *ACM Trans. Embedded Computing Systems*, vol. 9, no. 4, pp. 39:1–39:33, March 2010.
- [12] H. Møller, M. F. Sørensen, C. B. Jensen, and D. Hammershøj, "Binaural technique: Do we need individual recordings?" *J. Audio Eng. Soc.*, vol. 44, no. 6, pp. 451–469, June 1996.
- [13] E. A. G. Shaw and R. Teranishi, "Sound pressure generated in an external-ear replica and real human ears by a nearby point source," *J. Acoust. Soc. Am.*, vol. 44, no. 1, pp. 240–249, 1968.
- [14] P. Mokhtari, H. Takemoto, R. Nishimura, and H. Kato, "Frequency and amplitude estimation of the first peak of head-related transfer functions from individual pinna anthropometry," *J. Acoust. Soc. Am.*, vol. 137, no. 2, pp. 690–701, February 2015.
- [15] V. C. Raykar, R. Duraiswami, and B. Yegnanarayana, "Extracting the frequencies of the pinna spectral notches in measured head related impulse responses," *J. Acoust. Soc. Am.*, vol. 118, no. 1, pp. 364–374, July 2005.
- [16] S. Spagnol, M. Geronazzo, and F. Avanzini, "Fitting pinna-related transfer functions to anthropometry for binaural sound rendering," in *Proc. IEEE Int. Work. Multi. Signal Process. (MMSP'10)*, Saint-Malo, France, October 2010, pp. 194–199.
- [17] P. Bilinski, J. Ahrens, M. R. P. Thomas, I. J. Tashev, and J. C. Platt, "HRTF magnitude synthesis via sparse representation of anthropometric features," in *Proc. 39th IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP 2014)*, Firenze, Italy, May 2014, pp. 4501–4505.
- [18] F. Grijalva, L. Martini, D. Florencio, and S. Goldenstein, "A manifold learning approach for personalizing HRTFs from anthropometric features," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 3, pp. 559–570, March 2016.
- [19] S. Spagnol and F. Avanzini, "Frequency estimation of the first pinna notch in head-related transfer functions with a linear anthropometric model," in *Proc. 18th Int. Conf. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, December 2015, pp. 231–236.
- [20] S. Spagnol, M. Geronazzo, D. Rocchesso, and F. Avanzini, "Synthetic individual binaural audio delivery by pinna image processing," *Int. J. Pervasive Comput. Comm.*, vol. 10, no. 3, pp. 239–254, July 2014.
- [21] S. Spagnol, "On distance dependence of pinna spectral patterns in head-related transfer functions," *J. Acoust. Soc. Am.*, vol. 137, no. 1, pp. EL58–EL64, January 2015.
- [22] S. Müller and P. Massarani, "Transfer-function measurement with sweeps," *J. Audio Eng. Soc.*, vol. 49, no. 6, pp. 443–471, June 2001.
- [23] V. R. Algazi, C. Avendano, and R. O. Duda, "Elevation localization and head-related transfer function analysis at low frequencies," *J. Acoust. Soc. Am.*, vol. 109, no. 3, pp. 1110–1122, March 2001.
- [24] S. Spagnol, M. Hiipakka, and V. Pulkki, "A single-azimuth pinna-related transfer function database," in *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, September 2011, pp. 209–212.
- [25] G. F. Kuhn, "Model for the interaural time differences in the azimuthal plane," *J. Acoust. Soc. Am.*, vol. 62, no. 1, pp. 157–167, July 1977.
- [26] S. Spagnol, E. Tavazzi, and F. Avanzini, "Distance rendering and perception of nearby virtual sound sources with a near-field filter model," *Appl. Acoust.*, vol. 115, pp. 61–73, January 2017.
- [27] D. S. Brungart and W. M. Rabinowitz, "Auditory localization of nearby sources. Head-related transfer functions," *J. Acoust. Soc. Am.*, vol. 106, no. 3, pp. 1465–1479, September 1999.
- [28] M. Geronazzo, S. Spagnol, and F. Avanzini, "A modular framework for the analysis and synthesis of head-related transfer functions," in *Proc. 134th Conv. Audio Eng. Soc.*, no. 8882, Rome, Italy, May 2013.



# PERFORMING WITH SOUND SAMPLE-CONTROLLED GLOVES AND LIGHT-CONTROLLED ARMS

**Justin Pecquet**  
Jazz Institute of Berlin (JIB)  
Universität der Künste  
Berlin  
justin.pecquet@gmail.com

**Fotis Moschos**  
Vocational Training  
Institute  
Athens  
fotmos@windowslive.com

**David Fierro**  
CICM EA 1572  
MUSIDANSE  
Paris8 University  
davidfierro@gmail.com

**Frank Pecquet**  
ACTE Institute  
Paris1 Pantheon-Sorbonne  
University  
fpecquet@univ-paris1.fr

## ABSTRACT

Interacting with media: The TransTeamProject (T3P) works on developing interactive gloves techniques - and other materials, with sound and/or visual samples. *Piamenca* continues the work developed in *Transpiano*<sup>1</sup> with a specific emphasis on visual content such as transforming sound into lights, in this case together with a strong vernacular inspiration (Flamenco). The T3P creative project is involved with art music - as opposed to commercial music - together with technical perspectives. After contextualizing the state of the art in the specific field of “body gesture technology”, this paper will explain how *Piamenca* relates to computers in a practical sense – methods and processes to produce media transformations (both audio and visual) - and will comment on their integration in terms of sound, music and audio-visual performance. It will finally demonstrate some ideas such as trans-music orientations with regard to enhancement theories in relation with the transhumanism movement [1].

## INTRODUCTION

Although the so-called “interactive glove technology” has already a long history, as such in the interactive design research, whether or not related to music at first<sup>2</sup> (even though initially experimented with sound<sup>3</sup>) but more generally concerned with designing performance protocol for controlling trans-media data [2] [3], T3P develops normative procedures within the public domain by its own, making use of available cost-efficient hardware (Arduino board, Flex-Sensor) and software (Arduino-Uno, Max-MSP, Ableton Live). Once different technologies were originally mentioned as earlier as 1986 [4], but more recently redesigned such as for making music controlled by hand gestures, “Musical glove”<sup>4</sup>, Mi.MU glove technology<sup>5</sup>, Leap motion technology<sup>6</sup>, such experimentations were so far never thought of as for making art music but rather and more or less for pop sound productions. Within new technological processes, T3P team is developing typical formal ideas such as enhancing music performance

gestures to extend traditional piano fingering techniques.

## CONCEPTS FOR DESIGNING SOUND

*Piamenca* is conceived with fragments of Andalusian culture – Buleria and Tarento dances. Music patterns are arranged as identifiable elements looped in sessions of specific content, chords, melodies, rhythms to convey “audible design” [5]. Time structures depends upon (re)cycling modes of repetition using or not available transformations - inversion, reverse, transposition ... *Piamenca* uses diverse technics to filter sound spectrum, either from Max-MSP software or Ableton effect module in realtime.

Besides a quest for renewing acoustical sources - operating on traditional instrument while performing samples, such experiment concerns enhancing traditional gestures in order to recover empathy and produce emotional feedback in acting the sound with the help of visual representation.

## INSTRUMENTAL CONTEXT

*Piamenca*, such as *Transpiano*, makes use of the piano as it concretizes in the written classical heritage the “king of the instrument” for accompaniment. Not to mention its leadership over the Romantic composers which even made piano music sound dictatorial, piano remains the perfect visual, plastic tool to conceptualize all types of music. Nonetheless, it is not the original musical instrument of the flamenco technique. But today each instrument may easily be replaced by any other instrument through digital sampling technics [6]. While a totally autonomous piano does not really connect with the acting body - although it does in blind MIDI performance since we can see the key performing activity<sup>7</sup>, and while the glove techniques might still limit any emotional response, sampling techniques can really apply to music production with specific theatrical embodiment when prerecorded by the performer himself as a re-appropriation of the initial feeling at a second stage.

<sup>1</sup> Premiered last year at Limassol, Cyprus - XV SMC Conference (2018)

<sup>2</sup> Power Glove from Nintendo for Video Controls (2005)

<sup>3</sup> MODO Music Technology

<sup>4</sup> <https://interactiondesign.sva.edu/projects/musical-glove>

Copyright: 2019 Justin Pecquet et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>5</sup> <https://mimugloves.com/>

<sup>6</sup> <https://www.leapmotion.com/>

<sup>7</sup> We would mention here the expressive os Trond Reinholdsten piano concerto (2016) named « The theory of the subject.»

## WHY *PIAMENCA* IS ANOTHER TRANS-PIECE ?

While *transpiano* was the initiated version of such a starter “chain” of more works to come, as a first experimented fragment for designing different performing technics in art music, including videos, *Piamenca* uses extended conceptual technics with lights. Two basic ideas are proposed in the piece: first to develop sample patterns that structure listening; second, to produce a related visual matter - light, beams and video in relationship with the patterns. The overall form depends upon mixed media with a strong vernacular flamenco identity for the sake of this specific event.

## TECHNOLOGICAL CONTEXT : SOUND CONTROL

It logically happens that the basic idea behind the technological part of the T3P project is roughly the same as that of a MIDI keyboard. The performer has ten sensors (one on each finger). The sensors send raw data to the Arduino [7]. Then the Arduino sends the data via Bluetooth to the computer, more precisely to the Max-MSP. Max-MSP recognizes and separates this data from the Arduino (which have a range of 0-1023) and converts them directly to MIDI (range 0-127). A general setting of the performance is shown on fig. 1.

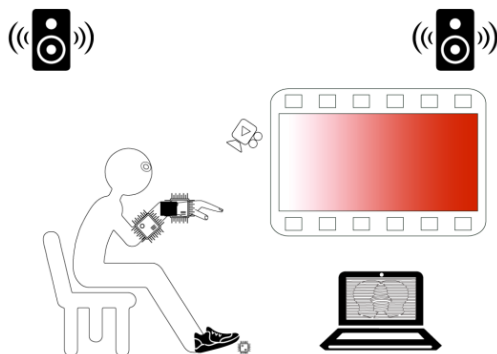


Figure 1. Elements for the setting

## FORMALIZING ELECTRONIC SOUND DATA

With such an interactive situation one may first resolve the fact that the performer has only ten fingers but might use more parameters to control (over 150 parameters). The following list mentions some of the technical issues to be resolved in such a project, from a sound point of view:

- Trigger the correct samples;
- Control the score;
- Enable/disable different effects;

- Control the correct parameters of all effects (some lists more than 50 parameters), etc.

## MAPPING: MAXMSP TO ABLETON LIVE

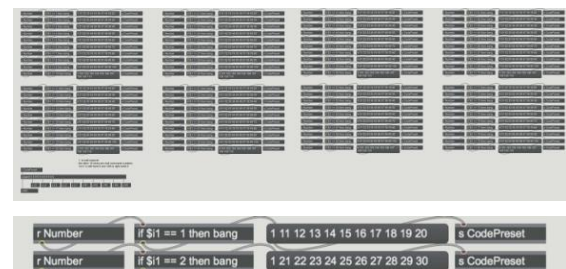
On a real piano, the performer controls pitches by pressing keys with fingers. In this project pitches (or sounds in extended technics) are sampled<sup>8</sup>. In this environment, fingers do not produce pitches but rather trigger samples and control different effects that would apply to them. Moreover, the performing activity is as follows: first, the performer selects a preset number with a foot controller - each preset already matches one or more samples; second, he triggers them with his thumbs to activate the sound(s) - at the same time, the other eight fingers are associated with the correct effect parameters already designed; third, he performs music with his fingers - instead of controlling pitches, the performer controls and transforms samples according to preconceived order impacting their own structural composition. A detailed score provides information for changes, starting over by selecting a new preset number and so on (Fig. 2).



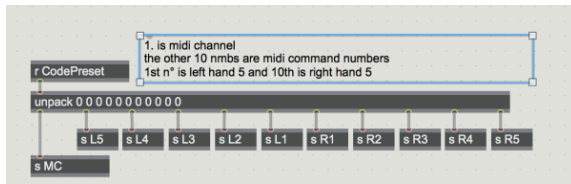
Figure 2. Diagram of the Preset subpatch

## PRESETS SOLUTION

In order to divide the piece into sessions in Ableton Live we used the “master session” command in Ableton with subsequent sessions to be triggered linearly. Each master session would correspond to a unique line in the program, a unique sample, a unique effect and in some cases, pages in the score. So, this particular display allowed to create one preset (or more if needed) for each session. The performer would select the correct preset with a foot controller. Each preset has ten MIDI CC, one for each finger. The MIDI protocol has 16 channels and each channel has 127 MIDI CC, so one may use up to 2032 MIDI CC or 2032 controllable parameters! (Fig. 3).



<sup>8</sup> A sample refers here to recorded precomposed sounds to be modified during the performance, never mind they be pitch sequences or “sonorities”.



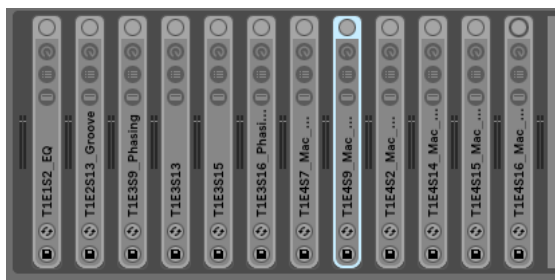
**Figure 3.** Preset structure

## LIMITATION OF MIDI CC

Although Ableton is perhaps the software with the easiest MIDI mapping system - we would have however some recommendations for improvement -, it is impossible to have two different MIDI CC connected to the same parameter. In other words, it was impossible to connect the same effect, let's say a basic Reverb, with two or even more different samples because each sample has a distinct preset – MIDI CC. This obstacle was resolved somehow in an unorthodox way. Instead of having one effect for all the samples, the same effect was used as many times as needed for the samples. For example, in the case of using three samples with a reverb, the same reverb was used three times. With such a setting it is therefore possible to assign one MIDI CC for each sample and each effect.

## IDENTIFYING EFFECT PARAMETERS

Beyond the number of effect parameters, the performer challenge is also to master their efficiency [8]. To this extent, one needs to change and provide a unique name for each effect, which is also combined with the dedicated score to easily identify sound properties as well as fingering functional settings. Instead of naming the followings “reverb1”, “reverb2”, “reverb3”, better use such codes T2E4S7, T3E1S8, T3E1S9 where T stands for Track, E for Effect and S for Session (Fig. 4)



**Figure 4.** Ableton Live Effects accessible from a single track (T1)

In the above example, the effect T1E1S2 means Track 1, Effect 1 and Session 2. With such a technique the performer knows exactly which effect was related to which sample. At the same time, the same preset enables only the correct effect(s) and selects them so that whenever the performer triggers a sample, he only sees the related effects in Ableton's window.

To resume this section, such a project raised technical obstacles related to the interface design but not with the glove and its shelf, neither with the Arduino board and the connecting system. Composing with such environment brings to the fore weaknesses in the design of predefined software (Ableton's GUI), due to its original conception, in this case processing samples in realtime. Technically, it might be unorthodox to have an exact same effect multiple times (calling for more RAM memory, resulting in lower compiler performance), with all 150 parameters, but it was the only solution available to be able to identify and control them with ten lively fingers!

## LIGHT CONTROL USING ARM SENSORS

In order to make a full mediation between different types of media, *Piamenca* also explores the relationship between sound and light, emphasizing performance strategies for the musician, such as mixing light properties with regard to acoustic correspondence controlled by specific body movements<sup>9</sup>. Using light intensities, color spectrum, shadows and interaction between glowing features and objects are part of demonstrating sound matter in this interactive process<sup>10</sup>.

As colors may change the way one perceives sound [9], we face this issue in three different perspectives:

- Sound analysis
- Musician interaction.
- Music composition.

As sound and light are waves stimulating our senses in distinctive frequency spectra, *Piamenca* explores this relationship using some previous studies on the subject. Amidst several stimulating data found and longtime analyzed is the impressive concordance that “synesthetic people” have in relationship with colors and musical notes [10].

In order to translate sound spectrum into light spectrum we are considering the sound produced by the musician to have the frequencies that compose the sound. By using the Fourier transform we analyze the sound spectrum in real time and use that information as input to the software which controls the lighting system.

While the performance is taking place, the musician is “touching” the music with his fingers - or according to arm gestures - by sending information to the system with absolute position. This project also allows fingers and arm positions to control the way light is projected and how the sound spectrum is transformed into lights and colors. As every performance would depend upon specific objectives due to artistic motivation, there are numerous ways for sound to be converted into light. Each composition requires its own choices. For the *Piamenca* project, we chose a light spectrum close to

<sup>9</sup> To this regard, let's mention the PIGS (Percussive Image Gestural System) prototype from Amy Alexander.

<sup>10</sup> Reference to be made here to the seminal work of Waisvisz M from STEIM. <http://www.crackle.org/TheHands.htm>

the red tonality in order to represent the distinctive red dress of the flamenco dancers.

## TECHNICAL VISUAL INSTALLATION

The light system used for this performance requires one led strip of five meters and four led light spots. The whole system is controlled by an Arduino card which receives the information coming from the software by the USB port. As the led strip shape can be modified, we use it to interact with the space to generate shadows and light movements. The four directional led lights are placed behind the musician in order to play with the space perceived by the audience.

## PERFORMANCE MEDIA CONTROLLED BY BODY GESTURES

At the root of digital composition, and in our peculiar context, sampling techniques for performance means expressing the self with “automated content”. The performer has some cues, since he is working on prerecorded sequences that he performed himself. However, questions on performing arise such as automated transforming samples and/or how looping patterns may influence decision making? While automation describes musical material depending upon audio data streaming, processes develop sound material and generate self-determination choices besides a more confined frame. In effect one cannot say it is pure improvisation while data lies on a score [11]. However, within a time frame process launched by gestures, it covers a wide range of body interaction that links to connected control. A set of constraints allows this type of musical processing [12].

## CONCLUSIONS

Although the T3P team’s philosophy isn’t limited to traditional musical instruments but would extend to other types including non-musical “objects” and media, *Piameuca* uses a piano sound without a physical body. The digital move of this century opens the way to further technological involvements with efficient materials such as sensors, microcontrollers, software, controlling systems, all easily affordable (and usable) outside institutions, freeing somehow creation from normative cultural appreciations. Not only such an opportunity saves the instrument, it also modifies the sound perception - processing, listening and visualizing, by creating a virtual gap between performing and producing sound in relation to other media, innervating space with lights and finally interacting all together. As transhumanism is also, amongst other theories, a way to consider technics in relationship to human, the so-called “anthropological technicity”, in art culture - where music is thought of as a voice to express ideas, it is also a matter of “trans” actions related to different contexts - socio-economic, ethical & political, neuro-bio-GIA-scientific, esthetic & artistic -. From these different angles performing art allows different positions, fundamentally based on mediating strategies for multimodal awareness.

## REFERENCES

- [1] Damour Franck, Deprez Stanislas, Daat David, Généalogies et nature du transhumanisme : Etat actuel du débat. “Genealogy and nature of transhumanism: current state of the debate”. French Montreal, Liber: 2018, 198 pages.
- [2] Serafin Stefania, Perner-Wilson Hannah, Trento Stefano, Madgwick Sebastian, Grani Francesco, Mitchell Tom: “Controlling Physically Based Virtual Musical Instruments Using The Gloves.” Proceedings of the New Interfaces for Music Expression, London, 2014, pp. 44-56.
- [3] Lai Chi-Hsia, Tahiroğlu Koray: « A Design Approach to Engage with Audience with Wearable Musical Instruments: Sound Gloves.” Proceedings of New Interfaces for Music Expression, Ann Arbor, Michigan, USA, 2012, pp. .
- [4] Holland Simon, Willkie Katie, Mulholland Paul, Seago Allan, Music and Human-Computer Interaction. Springer Science & Business Media: 2013, 292 pages.
- [5] Wishart, Trevor, Audible design. A plain and Easy introduction to Practical Sound Composition. York: Orpheus the Pantomime: 1994, 139 pages.
- [6] Leymann, Harry, The digital revolution in music. French translation from German. Alia, Paris: 2017, 222 pages.
- [7] Steiner Hans-Christoph, “Firmata: Towards making microcontrollers act like extensions of the computer”. Interactive Telecommunications Program, New York University: 2009, pp. 125-130.
- [8] Rosenboom, David, “Propositional Music: On emergent properties in Morphogenesis and the evolution of Music: Essays, Propositions, Commentaries, Imponderable Forms, and Compositional Methods”. In Arcana: Musicians on Music, edited by J. Zorn, New York: Granary Books, 2001, pp. 203-222.
- [9] O'Regan J. Kevin, Why Red Doesn't Sound Like a Bell: Understanding the Feel of Consciousness. Published by Oxford University Press: 2011, 211 pages.
- [10] Cytowic E. Richard, Synesthesia: A Union of the Senses. MIT Press: 2002, 394 pages.
- [11] Dean Roger, Hyperimprovisation: Computer-Interactive Sound Improvisation. The computer music and Digital Audio Series, Volume 19: 2003, 203 pages.
- [12] Gelineck Steven, Böttcher, Niels: “6to6Mappr: An Educational Tool For Fast And Easy Mapping Of Input Devices To Musical Parameters.” Proceedings of the Audio Mostly Conference, Corfu, Greece, 2012, pp. 117-123.



# Melody Identification in Standard MIDI Files

**Zheng Jiang**  
Carnegie Mellon University  
zjiang1@andrew.cmu.edu

**Roger B. Dannenberg**  
Carnegie Mellon University  
rbd@cs.cmu.edu

## ABSTRACT

Melody identification is an important early step in music analysis. This paper presents a tool to identify the melody in each measure of a Standard MIDI File. We also share an open dataset of manually labeled music for researchers. We use a Bayesian maximum-likelihood approach and dynamic programming as the basis of our work. We have trained parameters on data sampled from the million song dataset [1, 2] and tested on a dataset including 1703 measures of music from different genres. Our algorithm achieves an overall accuracy of 89% in the test dataset. We compare our results to previous work.

## 1. INTRODUCTION

When we listen to a piece of music, the melody is usually the first thing that catches our attention. Therefore, the identification of melody is one of the most important elements of music analysis. Melody is commonly understood to be a prominent linear sequence of pitches, usually higher than harmonizing and bass pitches. The concept of melody resists formalization, making melody identification an interesting music analysis task. Melody is used to identify songs. Often, other elements such as harmony and rhythm are best understood in relation to melody.

Many music applications depend on melody, including Query-by-Humming systems, music cover song identification, emotion detection [3], and expressive performance rendering. Many efforts in automatic composition could benefit from training data consisting of isolated melodies.

There has been a lot of research on extracting melody from audio [4]. The problem is generally easier for MIDI than audio because at least notes are already identified and separated. However, compared to audio, there seems to be less research on melody extraction. Most of the research on MIDI melody is on channel-level identification. This paper will propose an algorithm combining Bayesian probability models and dynamic programming to extract melody at the measure level.

## 2. RELATED WORK

In the field of symbolic files, *Skyline* is a very simple algorithm proposed by Uitdenbogerd [5]. In brief, the idea

of this method is to pick the highest pitch at any moment as belonging to the melody. Chai and Vercoe offer an enhanced version of this approach [6]. In pop music, we observe that there are often accompaniment notes above the melody line, leading to failure of the *Skyline* algorithms. Uitdenbogerd presents three more methods [4]: 1) Top Channel (choose the channel with the highest mean pitch), 2) Entropy Channel (choose the channel with the highest entropy), and 3) Entropy Part (segment first, then use Entropy Channel). Shan [7] proposed using greatest volume (MIDI velocity) because melody is typically emphasized through dynamics. Li et al. identify melodies by finding common sequences in multiple MIDI files, but this obviously requires multiple versions of songs [8]. Li, Yang, and Chen [9] use a Neural Network and features such as chord rate, pronunciation rate, average note pitch, instrument, etc., trained on 800 songs to estimate the likelihood that a channel is the melody channel. Velusamy, et al. [10] use a similar approach, but prune notes that do not satisfy certain heuristics and use a hand-crafted linear model for ranking channels [9]. Rizo, et al [11] introduces an algorithm to identify the track that contains the melody using statistical properties of the musical content and machine learning techniques.

All of these algorithms assume that the melody appears on one and only one channel, so the problem is always to select one of up to 16 channels as the melody channel. Depending on the data, this can be a frequent cause of failure, since the melody can be expressed by different instruments in different channels at different times. An interesting approach is *Tunerank* [12], which groups and labels notes according to harmony and dissonance with other notes, pitch intervals between consecutive notes, and instrumentation, without assuming the melody is in only one channel.

Previous work is hard to evaluate based on publications, with accuracy reports ranging from 60% to 97%, no labeled public datasets, and few shared implementations. The properties of music arrangements and orchestrations in MIDI files can cause many problems. The simplest case, often assumed in the literature, is that the melody appears in one and only one channel. At least four more complex conditions are often found: 1) The melody is sometimes played in unison or octaves in another channel, 2) the melody switches from one instrument (channel) to another from one phrase or repetition to another, 3) the melody is fragmented across channels even within a single measure (this happens but seems to be rare), and 4) there are multiple overlapping melodies as in counterpoint, rounds, etc.

Copyright: © 2019 Zheng Jiang et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



### 3. DATASET

In most related work, published links to datasets have expired, so we collected and manually labeled a new dataset which contains the training data of 5823 measures in 51 songs and test data of 1703 measures in 22 songs. For each song in the training data, the melody is mostly all on one channel, which is labeled as such. (This made labeling much easier, but we had to reject files where the melody appeared significantly in multiple channels or there is no melody at all.) In the test data, the melody is not constrained to a single channel, and each measure of the song is labeled with the channel that contains the melody. Measure boundaries are based on tempo and time signature information in the MIDI file. The MIDI files are drawn from songs in the Lakh dataset [1]. This collection contains MIDI files that are matched to a subset of files in the million song dataset [2]. We used tags there to limit our selection to pop songs.

The test data is collecting from Chinese, Japanese, and American pop songs. We specifically chose popular music because melody is usually present and there is usually a single melody. In addition, we hope to use this research in learning about melody structure in popular music.

It might be noted that there are many high quality MIDI files of piano music. Since all piano notes are typically on one channel, this can make the melody identification or separation a more challenging problem, and different techniques are required. We assume that in our data, once the channel containing the melody is identified, it is fairly easy to obtain the melody. Either the melody is the only thing present in the channel, or the melody is harmonized, and the melody is obtained by removing the lower notes using the Skyline algorithm.

### 4. ALGORITHM

Our problem consists of labeling each measure of a song with the channel that contains the melody. (It would be useful also to allow non-labels, or *nil*, indicating there is no melody, but our study ignores this option.) The algorithm begins with a Bayesian model to estimate  $M_{m,c}$ , the probability that channel  $c$  in measure  $m$  contains the melody. The estimation uses features that are assumed to be jointly normally distributed and independent. Features are calculated from the content of each channel, considering the measure itself and  $N$  previous and subsequent measures, with  $N \in 0, 1, \dots$ . In all of our experiments, we assume that melody *never* appears on channel 10, which is used for drums in General MIDI.

Although we could stop there and report the most likely channel in each measure,

$$c_m = \underset{c}{\operatorname{argmax}} M_{m,c} \quad (1)$$

this does not work well in practice. There are many cases where a measure of accompaniment, counter-melody or bass appears to be more “melody-like” than the true melody. (For example, the melody could simply be a whole note in some measures.) However, it is rare for

the melody to switch from one channel to another because typically the melody is played by one instrument on one channel. Channel switches are only likely to occur when the melody is repeated or on major phrase boundaries.

We can consider the melody channel for each measure,  $c_m$ , as a sequence of hidden states and per-measure probabilities as observations. We wish to find the most likely overall sequence  $c_m$  according to the per-measure probabilities, and taking into account a penalty for switching channels from one measure to the next. We model the probability of the hidden state sequence  $c_m$  as:

$$P(c_m) = \prod_m M_{m,c_m} S_{c_{m-1},c_m} \quad (2)$$

where  $S_{c_{m-1},c_m} = 1$  if there is no change in the channel ( $c_{m-1} = c_m$ ), and  $S_{c_{m-1},c_m}$  is some penalty less than one if there is a channel change ( $c_{m-1} \neq c_m$ ). Thus, channel switches are allowed from any measure to the next, but channel switches are considered unlikely, and any labeling that switches channels frequently is considered highly unlikely.

The parameters of this model must be learned, including: statistics for features used to estimate  $M_{m,c}$ , the best feature set, the number of neighboring measures  $N$  to use in computing features, and the penalty  $S$  for changing channels. We select the feature set and compute feature statistics using our training dataset, and we evaluate their performance and sensitivity to  $N$  and  $S$  using the test dataset.

#### 4.1 Bayesian Probability Model

The probability of melody given a set of features is represented by Equation 3, where  $C_0$  is the condition that the melody is present,  $C_1$  indicates the melody is not present,  $x_i$  are feature values, and  $n$  is the number of real-valued features. The details of features will be discussed in a later paragraph.

$$P(C_0|x_1, \dots, x_n) \quad (3)$$

By Bayes’ theorem, this conditional probability can be rewritten as Equation 4.

$$P(C_0|x_1, \dots, x_n) = \frac{P(C_0)P(x_1, \dots, x_n|C_0)}{P(x_1, \dots, x_n)} \quad (4)$$

With the assumption of independence for each feature, we can rewrite this as Equation 5:

$$P(C_0|x_1, \dots, x_n) = \frac{1}{Z} P(C_0) \prod_{i=1}^n P(x_i|C_0) \quad (5)$$

where  $Z$  is:

$$Z = P(x_1, \dots, x_n) = \sum_{k=0}^1 (P(C_k) \prod_{i=1}^n P(x_i|C_k)) \quad (6)$$

Our features  $x_i$  are continuous values. Because we have limited training data, we adopt a Naive Bayes approach and assume they are independent and distributed according

to a Gaussian distribution as in Equation 7. Under this assumption, we can simply collect feature statistics  $\mu_{i,k}$  and  $\sigma_{i,k}$  from training data to estimate the probability model.

$$P(x_i = v | C_k) = \frac{1}{\sqrt{2\pi\sigma_{i,k}^2}} e^{-\frac{(v - \mu_{i,k})^2}{2\sigma_{i,k}^2}} \quad (7)$$

We now describe the details of features, which are *note\_density*, *vel\_mean*, *vel\_std*, *pitch\_mean*, *pitch\_std*, *IOI\_mean*, and *IOI\_std*:

#### 4.1.1 Note Density

The note density is the sum of all note durations divided by the total length of the music (Equation 8). A melody without rests has a note density of 1, a rest has note density of 0, a sequence of triads without rests has a note density of 3, etc.

$$\text{note\_density} = \frac{\sum \text{note.dur}}{\text{total\_length}} \quad (8)$$

#### 4.1.2 Velocity

We take the mean and standard deviation of velocity (Equations 9 and 10).

$$\text{vel\_mean} = \frac{\sum_{i=1}^N \text{note}_i.\text{vel}}{N} \quad (9)$$

$$\text{vel\_std} = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (\text{note}_i.\text{vel} - \text{vel\_mean})^2} \quad (10)$$

#### 4.1.3 Pitch

We take the mean and standard deviation of pitch (Equations 11 and 12).

$$\text{pitch\_mean} = \frac{\sum_{i=1}^N \text{note}_i.\text{pitch}}{N} \quad (11)$$

$$\text{pitch\_std} = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (\text{pitch}_i.\text{vel} - \text{pitch\_mean})^2} \quad (12)$$

#### 4.1.4 Inter-Onset Interval

The Inter-Onset Interval, or IOI, means the interval between onsets of successive notes. Considering that ornaments and chords may introduce a very short IOIs, we set a window of 75 ms, and when two note onsets are within that window, we treat them as a single onset [13]. IOI calculation is described in detail in Algorithm 1.

## 4.2 Training data

We compute features for each measure and channel of the training data. For feature selection, we use cross-validation, dividing the training songs into 5 groups, holding out each group and estimating  $\mu_{i,k}$  and  $\sigma_{i,k}$  from the remaining training data, and evaluating the resulting model by counting the number of measures where the melody channel is judged most likely by the model. We take the average result over all five groups.

**Result:** The mean and standard deviation of a list of notes in onset-time order

*stats* is an object that implements the calculation of mean and standard deviation;

*note[i]* is the  $i^{\text{th}}$  note;

*N* is the number of notes;

*i*  $\leftarrow$  0;

**while** *i* < *N* **do**

*j*  $\leftarrow$  *i* + 1;

**while** (*j* < *N*)  $\wedge$  (*note*[*j*].on\_time - *note*[*j* - 1].on\_time) < 0.075 **do**

*j*  $\leftarrow$  *j* + 1;

**end**

**if** *j* < *N* **then**

*IOI*  $\leftarrow$  *note*[*j*].on\_time - *note*[*i*].on\_time;

*stats.add\_point*(*IOI*);

**end**

*i*  $\leftarrow$  *j*;

**end**

*IOI\_mean*  $\leftarrow$  *stats.get\_mean*();

*IOI\_std*  $\leftarrow$  *stats.get\_std*();

**Algorithm 1:** IOI Feature Calculation

After doing this for every combination of features (7 features, thus 127 combinations), for various values of *N* (the maximum distance to neighboring measures to use in feature calculation), we determine the features that produce the best result for each value of *N*.

We then re-estimate the probability model using *all* of the training data. In principle, we should also use the training data to learn the best window size *N* and the best penalty *S*, but in our training data, melodies are all in one channel, so the ideal value of *N* for this data should be large, and the ideal *S* should be zero (highest penalty) to prevent the melody from changing channels. Instead, we will determine *N* and *S* from our test data, where every measure is labeled as melody or not, and we will report how these parameters effect accuracy using the test dataset.

## 4.3 Melodic probability

To prepare for the dynamic programming step, we compute  $M_{m,c}$ , which is the natural log of the probability of melody in channel *c* at measure *m*. (The feature values are different for each combination of *m* and *c*.) In the next step, note that if we find the labels with the greatest sum of log probabilities, it is equivalent to finding the labels with the greatest product of probabilities. Logarithms are used to avoid numerical underflow.

## 4.4 Dynamic Programming

We use dynamic programming to select the channel containing the melody in each measure. Algorithm 2 shows how the assignment of channels maximizes the sum of  $M_{m,c}$  values adjusted by subtracting  $SP = -\log(S)$  each time the melody changes channels. The backtracking step is not shown since it is standard.<sup>1</sup>

<sup>1</sup> [https://en.wikipedia.org/wiki/Viterbi\\_algorithm](https://en.wikipedia.org/wiki/Viterbi_algorithm)

**Result:** For each measure, determine the channel containing the melody.  
 $N$  is the number of measures, indexed from 0 to  $N - 1$ ;  
 $C$  is the number of channels, indexed from 0 to  $C - 1$ ;  
 $SP$  is channel switch penalty, a parameter;  $SP = -\ln(S)$ ;  
 $A_{m,c}$  is the accumulated score for measure  $m$  and channel  $c$ ;  
 $B_{m,c}$  stores the optimal channel number of previous measure;  
 $M_{m,c}$  tells how melodic is channel  $c$  in measure  $m$ ;  
**for**  $i$  in  $[0 \dots C)$  **do**  
     $A_{0,i} \leftarrow M_{0,i}$ ;  
**end**  
**for**  $m$  in  $[0 \dots N)$  **do**  
    **for**  $c$  in  $[0 \dots C)$  **do**  
         $x \leftarrow A_{m-1,c} + M_{m,c}$ ;  
         $B_{m,c} = c$ ;  
        **for**  $i$  in  $[0 \dots C)$  **do**  
             $y \leftarrow A_{m-1,i} + M_{m,c} - SP$ ;  
            **if**  $c \neq i \wedge y > x$  **then**  
                 $x \leftarrow y$ ;  
                 $B_{m,c} \leftarrow i$ ;  
            **end**  
        **end**  
         $A_{m,c} \leftarrow x$ ;  
    **end**  
**end**

**Algorithm 2:** Dynamic Programming

## 5. EXPERIMENTS AND RESULTS

### 5.1 Training

We tried different combinations of features, and the results are shown in Table 1 for windows with 5 measures ( $N = 2$ ). The top 5 feature sets are shown along with the mean and standard deviation of accuracy across 5-fold cross-validation. Differences among the top feature sets are minimal. We use all features except velocity standard deviation.

Table 2 shows the results using each feature individually for 5-measure windows. This shows that all features offer some information (random guessing would be 1/15 or less than 7% correct), but no single feature works nearly as well as the best combination.

If we assume the melody appears in only one channel, which is mostly the case for this training dataset, we can consider the measure-by-measure melody channel results as votes, picking the channel with the majority of votes as the melody channel. Our best feature set (all but velocity standard deviation) gives an accuracy of 96% (2 errors out of 51 songs), using 5-fold cross-validation. In the next section, we relax the assumption that the melody appears in only one channel.

### 5.2 Testing

Our test dataset labels each measure with a set of channels containing melody. In measures with no melody, this is the empty set. In some measures, the melody is duplicated

nd	pm	ps	im	is	vm	vs	mean	std
1	1	1	1	1	1	0	72.40%	7.20%
1	1	0	1	1	1	0	71.60%	7.06%
1	1	1	1	1	1	1	71.40%	8.26%
1	1	1	1	0	1	0	71.40%	7.70%
1	1	1	1	0	1	1	71.00%	8.83%

Table 1. Mean and standard deviation of accuracy in 5-fold cross validation using the top 5 feature sets, window size = 5. Here, nd means *note\_density*, pm means *pitch\_mean*, ps means *pitch\_std*, im means *IOI\_mean*, is means *IOI\_std*, vm means *vel\_mean*, and vs means *vel\_std*.

nd	pm	ps	im	is	vm	vs	mean	std
1	0	0	0	0	0	0	45.80%	7.22%
0	1	0	0	0	0	0	44.60%	8.11%
0	0	1	0	0	0	0	35.80%	6.50%
0	0	0	1	0	0	0	31.00%	2.55%
0	0	0	0	1	0	0	30.40%	5.64%
0	0	0	0	0	1	0	32.00%	5.87%
0	0	0	0	0	0	1	20.60%	4.10%

Table 2. Mean and standard deviation of accuracy in 5-fold cross validation using individual features, window size = 5

in different channels, so the label can contain more than one channel. Note that if we can identify one channel containing the melody, it is simple to search for copies in the other melodies. Our algorithm labels *every* measure with exactly one melody channel. We consider the output to be correct either if it is in the set of true melody channels according to our manual labels, or if the label is the empty set. Typically, the empty set (no melody label) appears in introductions, endings, and measures where the melody channel rests. In these cases (approximately 12% of all measures), there is no clearly correct answer, so we will not include that in the test.

We evaluated accuracy on the test dataset with many values of  $N$  and  $SP$ . For each value of  $N$ , we used the best feature set as determined from the training data and then evaluated the system with different values of  $SP$ . The results are shown in Figure 1.

Since  $N$  and  $SP$  are optimized on the test dataset to obtain a best accuracy of 89.15%, there is some risk of overfitting parameters to the test data. Given more labeled data, we would have used a different dataset to select  $N$  and  $SP$ , and then we could evaluate the entire system on the test dataset. Instead, we argue that the system is not very sensitive to  $N$  or  $SP$ , so overfitting is unlikely. Figure 2 shows how accuracy is affected by varying the window size using an optimal value of  $SP = 36$  (again, the window includes the measure  $\pm N$  measures, so the window size is  $2N + 1$ ). This figure shows that 5-measure windows worked the best, but windows up to about 15 measures also work well, with just a few percent variation in accuracy.

Figure 3 shows how the accuracy is affected by varying  $SP$ , the switch penalty, using the optimal value of  $N = 2$ .

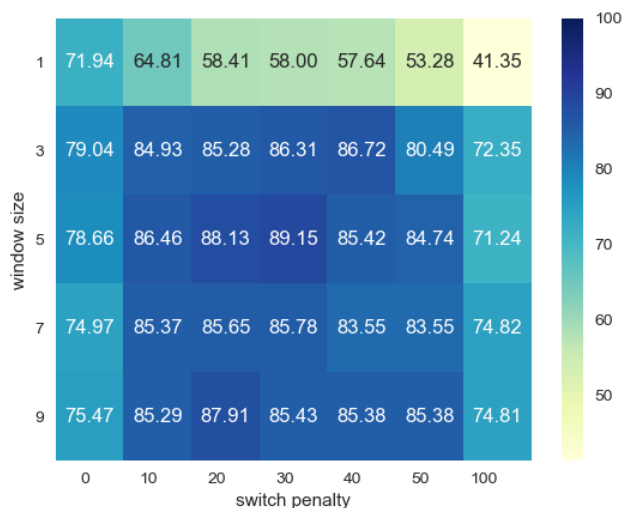


Figure 1. Accuracy for different values of window size and switch penalty.

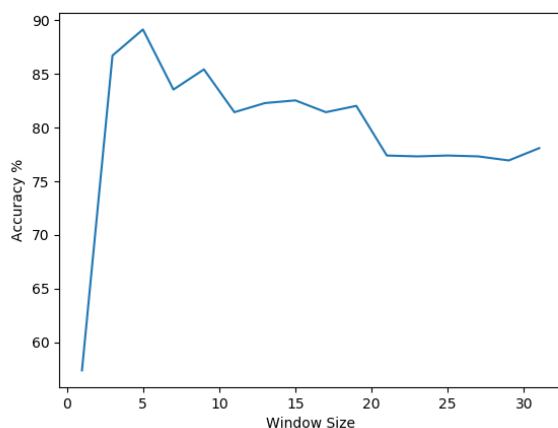


Figure 2. Accuracy on the test dataset vs. Window Size ( $= 1 + 2N$ ) for Switch Penalty = 36.

The best performance is obtained with  $SP$  between 30 and 38, but any value from 2 to 38 will achieve performance within a few percent of the best. Since both graphs are fairly flat around the best values of  $N$  and  $SP$ , the exact values of these parameters are not critical for good performance. In fact, we would expect the best values may depend upon style, genre, and other factors.

From the results, we can observe that the increase of window size helps the performance. The highest accuracy goes from 58.00% to 89.15% when the window size grows from 1 to 5. However, accuracy does not continue to increase for even larger windows. We believe the size of 5 measures is large enough to obtain some meaningful statistical features yet small enough to register when the melody has switched channels. In the next section, we analyze some specific examples of success and failure. We also see that the switch

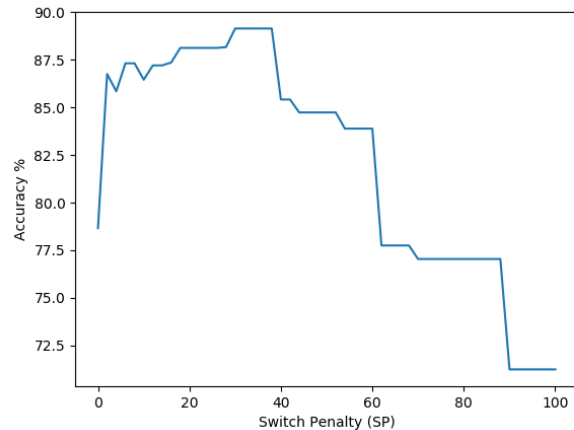


Figure 3. Accuracy on the test dataset vs. Switch Penalty with Window Size = 5. The highest accuracy is 89.15% using any Switch Penalty  $\in \{30, 32, \dots, 38\}$ .

penalty matters. When we set the penalty to zero, we get the locally best choice of melody channel at each measure, independent of other measures, but it seems clear that this “locally best, independent” policy is not particularly good, and this is why we introduced the Viterbi step to our algorithm. On the other hand, when the penalty is very large, we force all measures to be labeled with the same channel. This is not a good policy either, with at best 71.24% accuracy. The results show that our Viterbi step is effective in using context to improve melody identification.

## 6. ANALYSIS

To better understand our approach, we analyzed some songs in our test dataset.

### 6.1 A Successful Sample

In most of the cases, this algorithm works well. For example, the figure 4 shows a clip from the popular song “Hotel California.” In the figure, the melody is labeled by our algorithm in red (at the top) and other notes are shown in yellow. Notice that this melody is not particularly “melodic” in that it only uses two pitches and there is a lot of repetition. This is one illustration of the need for multiple features and statistical methods. The features for the melody channel have a higher likelihood according to our learned probabilistic model, and the melody is correctly identified.

### 6.2 Failed Samples

A failure case is shown in Figure 5. Here, the detected melody is shown in red at the top of the figure. The same (red) channel actually contained the melody in the immediately preceding channels, but at this point the melody switched to another channel, shown below in yellow. (In the figure, the lower melody is visually separated for clarity, but both channels actually occupy the same pitch range.) Evidently, the algorithm continued to label the top





Figure 4. The melody detected in the song “Hotel California.”

(red) channel as melody to avoid the switch penalty that would be required to label the melody correctly. In fact, the true (yellow) melody appeared earlier in the top (red) channel, so perhaps a higher-level analysis of music structure would also be useful for melody identification and disambiguation.



Figure 5. The algorithm identified the top (red) channel as melody of “Being,” but the correct melody is shown in yellow at the bottom.

Another song in our dataset is “Ali Mountain.” In this piece, at measure 12, the melody is split across two different channels, represented in red (darker) and yellow (lighter) in Figure 6. Taken together, the combined channels would be judged to be very melodic. However, when we consider the channels separately, it is hard to hear whether either is part of a melody, and our algorithm does not rate either channel highly. Since we assume that the melody will be played by one and only one channel within a measure, the melody is not identified in this test case.



Figure 6. Two channels ensemble the melody

## 7. CONCLUSION

In this paper, we contributed a novel algorithm to detect the melody channel for each measure in a MIDI file. We utilize a Bayesian probability model to estimate the probability that the melody is on a particular channel in each measure. We then use dynamic programming to find the most likely channel for melody in each measure considering that switching channels from measure to measure is unlikely. We obtained an overall accuracy of 89% on our test dataset, which seems to compare favorably to most other results in the literature. The lack of a large shared dataset prohibits a detailed comparison.

Our dataset, including Standard MIDI Files, melody labels, associated software, and documentation are available at the following website:

<http://www.cs.cmu.edu/~music/data/melody-identification>.

## 8. FUTURE WORK

We believe further improvements could be made by studying failures. With bootstrapping techniques, it might be possible to obtain much more training data and learn note-by-note melody identification, which would solve the problem of melodic phrases split across two or more channels. Our current dataset is relatively small, so collecting a larger dataset could be beneficial for tuning this algorithm and developing others. With larger datasets, deep learning and other techniques might be enabled. Perhaps bootstrapping (or semi-supervised learning) techniques could be used starting with the present algorithm to label a larger dataset automatically. We also believe that music structure can play an important role in melody identification. Melodies are likely to be longer sequences that are repeated and/or transposed, and these non-local properties might help to distinguish “true” melodies as perceived by human listeners, even when the melodies are not particularly “melodic” in terms of local features.

## Acknowledgments

In this work, we would like to acknowledge Shuqi Dai for providing some test samples in our dataset.

## 9. REFERENCES

- [1] C. Raffel, *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. Columbia University, 2016.
- [2] B.-M. Thierry, P. E. Daniel, W. Brian, and P. Lamere, “The million song dataset,” in *ISMIR 2011: Proc. the 12th International Society for Music Information Retrieval Conference, October 24–28, 2011, Miami, Florida*. University of Miami, 2011, pp. 591–596.
- [3] Z. Wei, L. Xiaoli, and L. Yang, “Extraction and evaluation model for the basic characteristics of midi file music,” in *Control and Decision Conference (2014 CCDC), The 26th Chinese*. IEEE, 2014, pp. 2083–2087.

- [4] C. Isikhan and G. Ozcan, "A survey of melody extraction techniques for music information retrieval," in *Proceedings of 4th Conference on Interdisciplinary Musicology (SIM08)*, Thessaloniki, Greece, 2008.
- [5] A. Uitdenbogerd and J. Zobel, "Melodic matching techniques for large music databases," in *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*. ACM, 1999, pp. 57–66.
- [6] W. Chai and B. Vercoe, "Melody retrieval on the web," in *Multimedia Computing and Networking 2002*, vol. 4673. International Society for Optics and Photonics, 2001, pp. 226–242.
- [7] M.-K. Shan and F.-F. Kuo, "Music style mining and classification by melody," *IEICE TRANSACTIONS on Information and Systems*, vol. 86, no. 3, pp. 655–659, 2003.
- [8] L. Li, C. Junwei, W. Lei, and M. Yan, "Melody extraction from polyphonic midi files based on melody similarity," in *Information Science and Engineering, 2008. ISISE'08. International Symposium on*, vol. 2. IEEE, 2008, pp. 232–235.
- [9] J. Li, X. Yang, and Q. Chen, "Midi melody extraction based on improved neural network," in *Machine Learning and Cybernetics, 2009 International Conference on*, vol. 2. IEEE, 2009, pp. 1133–1138.
- [10] S. Velusamy, B. Thoshkahna, and K. Ramakrishnan, "A novel melody line identification algorithm for polyphonic midi music," in *International Conference on Multimedia Modeling*. Springer, 2007, pp. 248–257.
- [11] D. Rizo, P. J. P. De León, C. Pérez-Sancho, A. Pertusa, and J. M. I. Quereda, "A pattern recognition approach for melody track selection in midi files." in *ISMIR*, 2006, pp. 61–66.
- [12] H. Zhao and Z. Qin, "Tunerank model for main melody extraction from multi-part musical scores," in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2014 Sixth International Conference on*, vol. 2. IEEE, 2014, pp. 176–180.
- [13] J. Bloch and R. B. Dannenberg, "Real-time accompaniment of polyphonic keyboard performance," in *Proceedings of the 1985 International Computer Music Conference*, 1985, pp. 279–290.

# AUTOMATIC CHORD-SCALE RECOGNITION USING HARMONIC PITCH CLASS PROFILES

**Emir Demirel**

Queen Mary University of London  
e.demirel@qmul.ac.uk

**Baris Bozkurt**

Izmir Democracy University  
barisbozkurt0@gmail.com

**Xavier Serra**

Universitat Pompeu Fabra  
xavier.serra@upf.edu

## ABSTRACT

This study focuses on the application of different computational methods to carry out a "modal harmonic analysis" for Jazz improvisation performances by modeling the concept of *chord-scales*. The Chord-Scale Theory is a theoretical concept that explains the relationship between the harmonic context of a musical piece and possible scale types to be used for improvisation. This work proposes different computational approaches for the recognition of the chord-scale type in an improvised phrase given the harmonic context. We have curated a dataset to evaluate different chord-scale recognition approaches proposed in this study, where the dataset consists of around 40 minutes of improvised monophonic Jazz solo performances. The dataset is made publicly available and shared on [freesound.org](https://freesound.org). To achieve the task of chord-scale type recognition, we propose one rule-based, one probabilistic and one supervised learning method. All proposed methods use Harmonic Pitch Class Profile (HPCP) features for classification. We observed an increase in the classification score when learned chord-scale models are filtered with predefined scale templates indicating that incorporating prior domain knowledge to learned models is beneficial. This study has its novelty in presenting a first computational analysis on chord-scales in the context of Jazz improvisation.

## 1. INTRODUCTION

In this work, we perform an automatic analysis of the tonal harmonic context in monophonic improvisation performances specifically in the context of Jazz tradition. As proposed in [1], the cultural context in music should be considered when conducting computational musicological research. Here, we employ *chord-scales* as the unit for computational analysis of Jazz improvisation. This paper proposes a new approach for the retrieval of chord-scale information from Jazz improvisation performances. In our study, various methods are compared for the classification of chord-scale types based on Harmonic Pitch Class Profiles (HPCP) extracted from audio signals. Due to the lack of a dataset with monophonic improvised performances that has temporal labels for the chord-scale that is played, we have curated a new dataset, which is called *The Chord-scale*

*Dataset*. The performance of the proposed methods were tested on this dataset alongside with a baseline score. This work proposes a novel approach for the analysis of the harmonic content in Jazz improvisation, targeting chord-scales as their use in a specific Jazz style is an essential skill [2].

In musical context, a scale is a step-wise arrangement of pitch classes contained within an octave [3]. A Jazz player is expected to draw on everything he or she knows concerning these scales and their relationships with chords [4]. Clearly, the choice of chord-scales to be played within a certain harmonic context is very subjective and dependent on performers artistic decisions. For practical reasons, a limitation on the list of chord-scales to detect or estimate is considered in this work. The readers are encouraged to refer to the Scale Syllabus by Jamey Aebersold [5] for an extended set of chord-scales used in Jazz improvisation.

Intuitively, audio features that represent pitch or pitch class information of the musical content would be ideal for the classification of chord-scales. One approach could be performing chord-scale classification based on (musical) note tracks. However, this approach would require a highly accurate transcription of the improvised content to achieve a robust classifier and predicting accurate results. To skip the automatic transcription step, we employ Harmonic Pitch Class Profiles (HPCP) as features for classification and the estimation of the chord-scale type.

Three methods are proposed for the classification stage. First, a set of predefined binary chord-scale templates are used in a pattern matching strategy based on likelihoods. The second approach is an automatic classification pipeline using Support Vector Machines (SVM). Finally, we apply a similar pattern matching strategy with that of the first method, but replacing predefined templates with chord-scale models learned from labeled data using Gaussian Mixture Models (GMM). Alongside the performance of the proposed chord-scale recognition methods, the choice of different statistical features for classification is tested during experiments.

This paper is structured as follows. First, a big picture of this work is introduced. Then relevant domain knowledge and a review of the previous attempts for musical scale estimation are provided. Then, we introduce the Chord-Scale dataset. Fourth and fifth sections explain the feature preprocessing stages. Later the automatic classification of chord-scale types are explained. The experiments comparing the performances of the proposed classification algorithms and varying statistical features are given in Section

Copyright: © 2019 Emir Demirel et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

7. Then we conclude with discussions regarding the results of the experiments and possible future directions of this study.

## 2. BACKGROUND INFORMATION

Due to the high degree of multidisciplinary of the research area of Music Information Retrieval (MIR), it is essential to have a comprehensive outlook that comprises both *music theoretical* (or domain) knowledge and the advanced computational methods for *music modeling* when doing research for solving an MIR task. Our study for chord-scale recognition proposes combining methodological approaches from both musical and machine learning domains. In this section, we examine the musical theoretical concepts that motivated this work and computational methods to analyze certain musical concepts that are related to chord-scales.

### 2.1 The Chord-Scale Theory

The *chord-scale theory* is a method of mapping a list of scales to a list of chords and the theory aims to explain the inter-relationships between the chords and scale in the context of Jazz improvisation. For last few decades, jazz musicians use the approach of chord-scales when improvising over chord progressions. The essence of the chord-scale approach is if a chord is diatonic to a certain scale, than that scale can be used as a resource for creating melodic lines.

In traditional approach for tonal harmony, chords are built in three tones or triads. In Jazz tradition though, seventh degrees and additional color tones are commonly used, describing a chord or a chord progression with all its potential tonal possibilities. Chords are vertical structures of notes where the chord tones are separated by leaps while scales form a step-wise arrangement of notes [2]. Playing consecutive steps is much easier for playing fast and accurately which makes the chord-scale theory a preferable approach for improvisation among jazz musicians.

The list of chord-scales involved in this study for analysis are determined according to the content of Gary Burton's on-line course of Jazz Improvisation. In addition to the scales involved in the course, we consider other mother scales included in [6] that are commonly used in Jazz performances. The complete list of chord-scale mapping to be considered in this study can be seen from Table 1.

### 2.2 Chord-Scale Type Recognition

The task of automatic chord-scale recognition (or detection) from musical audio is relatively a new field of research in Sound and Music Computing (SMC) and has not been investigated as deeply as other MIR tasks like *automatic music transcription*, *chord recognition*, *musical similarity*, *style identification*, etc. However the retrieval of chord-scale related information has a potential to be beneficial for many MIR tasks and applications including the aforementioned titles.

Even though there is no well developed literature specifically focusing on the automatic chord-scale recognition

task, there are relevant research that study related musicological concepts to chord-scales. In [7], Weiss and Habryka proposes an algorithm for visualizing the tonal characteristics of classical audio recording through the concept of scales. In the second approach presented in the paper, the authors apply maximum likelihood estimation on audio frames using chroma features to estimate the scale-type.

Conceptually, chords and chord-scales are theoretically and practically related concepts in musical practice. The same may hold true for the recognition of both musical concepts as separate Music Information Retrieval (MIR) tasks. Both chords and chord-scales are musical harmonic structures that are defined in terms of pitch classes. This relevance of these concepts inspired our study to employ some of the common probabilistic approaches used for chord recognition. In [8], the authors study the effects of employing binary chord templates and probabilistic chord models, where they do not report a significant performance improvement of using learned chord models as opposed to predefined binary templates. In [9], the authors use chromas extracted in various ways with the goal of establishing timbre invariant feature vectors. For chord recognition, Gaussian Mixture Models are used as the pattern matching strategy.

## 3. DATASET

We have curated a new dataset for the context of the study presented in this paper. The *Chord-Scale Dataset* consists of 39 monophonic improvisation performances in 12 chord-scale types (Table 1) commonly used in Jazz improvisation. The recording device used during the data collection sessions is Zoom H-6. The total duration of the dataset is around 40 minutes and the recordings belong to either tenor saxophone or trumpet performances. Throughout each recording the tonic and the tonal harmonic context is kept constant. The musical phrases (or *motifs*) in each recording are annotated with their start and end times. In order to maintain a balanced dataset in terms of number of phrases per chord-scale type, we have included 4 additional tracks from *Jamey Aebersold Jazz, Volume 26: The "Scale Syllabus"* [5]. In total, there are 43 tracks and 236 data points, each representing a musical phrase. Hence, the chord scale recognition in our study is performed on the phrase level

The dataset is shared publicly on *freesound.org*<sup>1</sup>. The phrase onset and offset annotations can be found in the same repository with the code to generate and reproduce the experiments<sup>2</sup>. The annotation format in this dataset is similar to the chord progression annotation format proposed in [10]. For reproducibility purposes, we have made the frame-level HPCP features extracted from the recordings available in the *github* repository. Moreover, this dataset also includes instrument labels for each recording.

<sup>1</sup> <https://freesound.org/people/emirdemirel/packs/24075/>  
Pack ID: 24075

<sup>2</sup> <https://github.com/emirdemirel/Chord-ScaleDetection>



#### 4. FEATURE EXTRACTION

pitch class Profiles (PCP) [11] or chroma features [12] [13] have been a popular tonal representation of musical signals since their introduction around two decades ago. In our system, we have used Harmonic Pitch Class Profile (HPCP) features for the recognition of the chord-scale type from a musical phrase. HPCP features are a specific type of chroma features which are extracted via weighted mapping of harmonic peaks in the frame spectrum onto 12 pitch classes [14].

One may argue that monophonic musical signals can be conveniently transcribed and analyzed in symbolic domain for more precision and explainability. Such automatic transcription can be achieved in two steps: First, pitch-tracking would be applied on the improvised performance. Then, the pitch track would be transcribed into musical note level. The automatic music transcription procedure introduces much more complex level of computation. Thus, we propose to use Harmonic Pitch Class Profile features for the simplicity in conducting a subbranch of automatic harmonic analysis that our study focuses on. Given the chord-scale types included in this study being octave invariant sets of pitch classes, meaning that the relative octave differences between pitches that belong to the same pitch class does not influence the chord-scale type estimation proposed in this study. According to this consideration, using HPCP features does not introduce inefficacy for achieving the chord-scale recognition task.

##### 4.1 Preprocessing

First, the audio signal is filtered with inverted approximation of equal loudness curves in order to account for the non-linear perception of the spectra in the human auditory system [15]. The sampled and filtered audio signal is divided into series of analysis frames of size  $N_{frame}$  and hop size of  $N_{hop}$ . Then, each analysis frame  $x(n + l \cdot N_{hop})$  is multiplied with a "Hanning" window function  $w(n)$ , to obtain the windowed audio signal  $x_w(n)$ , where  $n = 0, 1, \dots, N_{frame} - 1$  and  $l$  indicates the number of the frame that is analyzed.

##### 4.2 Harmonic Pitch Class Profiles

Chromagrams or Pitch Class Profiles (PCPs) are widely used in many applications that aim to extract mid-level musical information from the audio signal, like automatic chord recognition key extraction [16], cover song identification [17] and such, since they were introduced in [11].

In principle, Harmonic Pitch Class Profiles [14] are modified versions of Pitch Class Distributions (PCDs) which are introduced in [11]. In principle, the HPCP extraction procedure applies a weighted mapping on spectral peaks detected on frame-based spectra to a finite number of pitch classes. In our context, we use 12 pitch classes as there are 12 pitch classes defined within one octave in equally the tempered tuning system.

In order to maintain tonic invariant HPCP features, the frame level HPCP vectors are constructed with respect to the tonic frequency of the improvised phrase, so that the

reference frequency  $f_{ref}$  (HPCP bin # 0) corresponds to the tonic frequency. The tonic frequency is computed using the following formula:

$$f_{ref} = f_{tuning} \cdot 2^{\frac{\delta k \cdot 100}{1200}} \quad (1)$$

where  $f_{tuning}$  is the global tuning frequency of the performance,  $\delta k$  is the distance in semitones between the keys of the tuning frequency  $K_{tuning}$  and the target key  $K$ .

##### 4.3 Post-Processing

The goal of the post-processing steps explained in this section is to prepare the feature data for the classification stage. In order to establish dynamic invariance in the feature vectors, each of the frame-based HPCP vectors are normalized with respect to a suitable norm. In our methodology, we employ *unitSum* ( $l - 1$ ) norm. By applying *unitSum* norm, we obtain relative weights of pitch classes in the frame-based HPCP vectors.

The pitch class mapping of spectral harmonic peaks causes artifacts on frame-level HPCP vectors. In order to minimize these artifacts, we only consider HPCP bin with the maximum value in the frame-level HPCP vectors and set the rest of the bins to zero (Equation 2). The resulting feature vectors are denoted as  $HPCP'(i)$ . This process is valid for our case since the analysis audio files are monophonic / one-instrument performances and it is expected to have only one dominant pitch class in the feature frames.

$$HPCP'(i) = \begin{cases} HPCP(i), & \text{if } i = \max_i HPCP(i) \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

We apply further processing on the frame-based chroma features in order to reduce the influence of noise and artifacts in the summarized features for classification. The logic of our noise removal method is as below:

$$\begin{aligned} \text{if } \arg\max(HPCP'(n)) \neq \arg\max(HPCP'(n \pm 1)) : \\ HPCP'(n) = 0 \\ k = 0, 1, 2, \dots, 11 \end{aligned} \quad (3)$$

The recordings in the data set are monophonic and the post-processed feature vectors have only one non-zero component that is the pitch class with maximum energy in the unprocessed vectors. The non-maximum pitch classes are discarded on frame level targeting to obtain an overall pitch class profile distribution that has less influence from the harmonic artifacts and noise.

#### 5. FEATURE SELECTION

The chord-scale recognition approaches proposed in this paper are performed on phrase-wise summarized HPCP features. Frame-level features are summarized into 2 statistical aspects: the mean and the standard deviation. For each pitch class (or bins in HPCP vectors), the mean and

standard deviations are calculated over the manually annotated phrase segments. Then, the summarized features are normalized on  $l_2$  norm.

In Figure 1, the summarized feature histograms are shown for both of *mean* and *standard deviation* features.

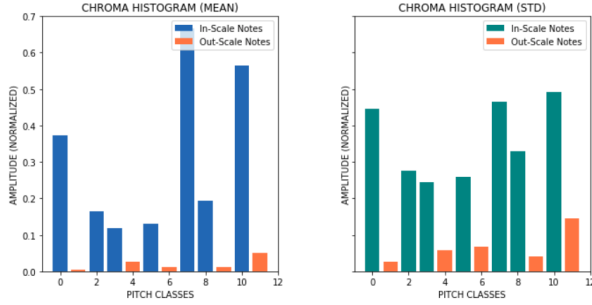


Figure 1: pitch class distributions of statistically summarized features. **Mean** features (*left*) and **std** features (*right*). Sample: 'Improvisation on Minor Scale - Trumpet - Phrase # 2'

It can be seen from the plots in Figure 1 that the histograms have non-identical but similar distributions over pitch classes, which indicates both features contain relevant information. Since these histograms represent statistical summarization of tonal features, it can be maintained that as both statistical features capture similar tonal aspects from the acoustic signal.

## 6. CLASSIFICATION

In this study, we propose 3 distinct chord-scale recognition algorithms. The first approach applies pattern matching using Maximum Likelihood Estimation (MLE) based on predefined binary chord-scale templates. In the second approach, we use Support Vector Machines (SVM) for classification of chord-scale types. Finally, we propose to replace binary templates in the first approach with chord-scale models that are learned using Gaussian Mixture Models (GMM).

### 6.1 Binary Template Matching

The first chord-scale recognition method we have developed follows a Binary-Template Matching (BTM) strategy, which is inspired by the scale matching method proposed in [7]. The method proposed is essentially a *maximum likelihood estimation* procedure where the likelihoods are computed as the product of pitch classes in the chroma vectors. We propose to compute the likelihoods as the sum of the pitch classes, due to the methodological and contextual differences explained in Section 4.3.

The chord-scale type likelihoods  $S$  are obtained by computing the inner products of the statistically summarized chroma vectors  $HPCP$  with each of the chord-scale templates  $T_s$ :

$$S(s) = \sum_{i=0}^{11} HPCP(i) \cdot T_s(i) \quad (4)$$

Table 1: Scale Dictionary

Scale Types $s$	Binary Templates $T_s$
Ionian (Major)	(1 0 1 0 1 1 0 1 0 1 0 1)
Dorian	(1 0 1 1 0 1 0 1 0 1 1 0)
Phrygian	(1 1 0 1 0 1 0 1 1 0 1 0)
Lydian	(1 0 1 0 1 0 1 1 0 1 0 1)
Mixolydian	(1 0 1 0 1 1 0 1 0 1 1 0)
Aeolian (Natural Minor)	(1 0 1 1 0 1 0 1 1 0 1 0)
Locrian	(1 1 0 1 0 1 1 0 1 0 1 0)
Melodic Minor	(1 0 1 1 0 1 0 1 0 1 0 1)
Lydian b7	(1 0 1 0 1 0 1 1 0 1 1 0)
Harmonic Minor	(1 0 1 1 0 1 0 1 1 0 0 1)
Altered (Super Locrian)	(1 1 0 1 1 0 1 0 1 0 1 0)
Whole Tone	(1 0 1 0 1 0 1 0 1 0 1 0)
Half-Whole Step Diminished	(1 1 0 1 1 0 1 1 0 1 1 0)

Finally, the chord-scale type  $s$  with the maximum likelihood would be determined as the estimated or detected scale type.

$$S' = \max_s S(s) \quad (5)$$

where  $S'$  denotes the final estimated chord-scale type.

### 6.2 Support Vector Machines

Support Vector Machine (SVM), first introduced in [18], is a classification and regression tool that uses a hypothesis space with linear functions in a high dimensional feature space, trained using a learning algorithm from optimization theory that implements a learning bias from statistical learning theory [19].

In our SVM classification model, we use statistically summarized feature vectors as the feature data for the classifier. The SVM classifier holds Radial Basis Function (RBF) kernels which provide more flexible mapping of feature spaces. The penalty parameter  $C$  and the kernel coefficient  $\gamma$  of the classifiers are optimized using Grid Search / Cross Validation as in [20]. The following parameter grids are iterated over to choose the best pair of hyperparameters:

$$C : \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$$

$$\gamma : \{0.001, 0.01, 0.1, 1\}$$

### 6.3 Gaussian Mixture Models

Finally, we propose a probabilistic approach that resembles the binary-template based likelihood strategy in Section 6.1 but applies pattern matching on learned chord-scale models from labeled data. In this approach, each chord-scale model is constructed using Gaussian Mixture Models (GMM) defined in terms of mean  $\mu$  and a covariance matrix. A GMM is a weighted sum of multivariate Gaussian distributions [21], which is defined as :

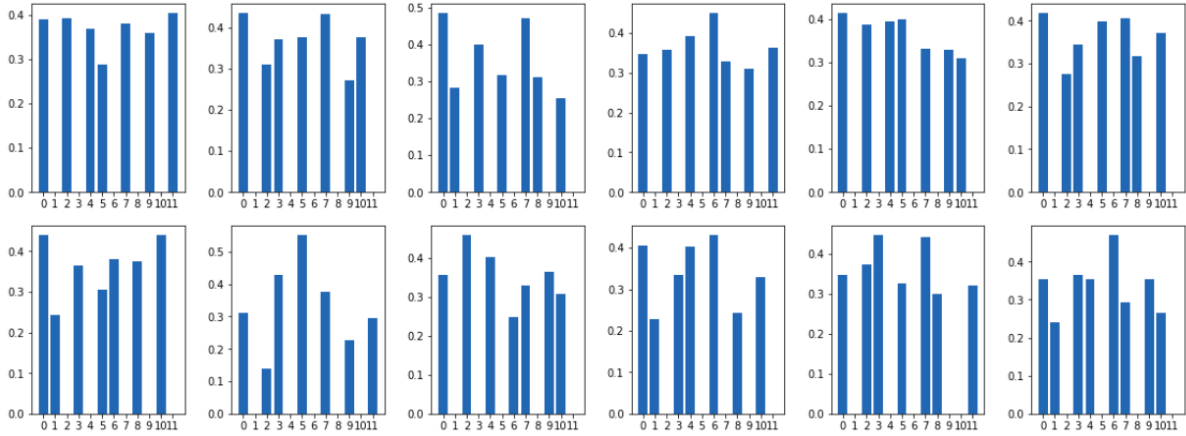


Figure 2: Chord-Scale Models learned using GMM,  $x$  axis = *HPCP index*,  $y$  axis = weights assigned by GMMs  
 Top row: *major, dorian, phrygian, lydian, mixolydian, minor*  
 Bottom row: *locrian, melodic minor, lydian b7, altered, harmonic minor, half-whole step diminished*

$$p(\mathbf{x}) = \sum_{k=1}^K c_k \frac{1}{\sqrt{|2\pi \sum_k|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_k)^T \sum_k^{-1} (\mathbf{x} - \mu_k)\right) \quad (6)$$

where  $K$  is the number of mixture components in the Gaussian distribution. In the case of  $K = 1$ , the distribution employs the maximum likelihood estimate of parameters. In our approach, we construct a GMM with one component ( $K = 1$ ) for each chord-scale type. Then the estimation of the likeliest chord-scale in Equation 4 is applied by replacing the binary templates  $T_s$  by  $\mu$ .

$$S(s) = \sum_{i=0}^{11} HPCP(i) \cdot \mu_s(i) \quad (7)$$

Finally, the chord-scale type with the maximum likelihood is determined (as in Equation 5) as the chord-scale estimate.

## 7. EXPERIMENTS

The experiments are conducted to test the choice of statistical summarization of features and the performance of the proposed algorithms for chord-scale recognition. The method introduced in [7] is also tested on the *Chord-Scale Dataset* as comparison with the proposed classification algorithms during evaluation.

The pattern matching algorithms (BTM and GMM) employ MLE for classification. MLE is applied on the mean vectors for each chord-scale model. Note that the out-scale pitch classes in GMM models have non-zero values. Since the MLE procedure in this study does not give penalty to out-scale pitch classes, these features potentially decrease the classification performance. In order to overcome this and obtain a higher performance, we apply element-wise multiplication on the binary-templates and the learned models, which filters out the out-scale pitch classes (setting them to zero) and eliminating the effect of these features on

MLE-based classification. The resulting chord-scale models are shown in Figure 2. The performances of both the filtered and unfiltered GMM models are tested during the experiments.

The implementations of machine learning algorithms used in this study are obtained from **scikit-learn**, which is a *Python* module that integrates a broad range of state-of-the-art machine learning algorithms for medium scaled problems [22].

## 8. RESULTS

As explained in Section 6.2, the hyperparameters of the SVM classifier are optimized using Grid Search. First, the dataset is split into two balanced subsets: train and test sets. 10-fold cross validation is applied on the train set for optimizing the hyperparameters of the SVM classifier. More explicitly, the train set is split in 10 folds and one fold is chosen as the validation set at each iteration. The validated hyperparameter pair is then used to evaluate the performance of the classifier on the test set. In order to increase the generalization power of this procedure, we classification scores are obtained 10 times on pseudo randomized development and test splits. Then, the average performance scores of all iterations are reported as the final SVM classifier score. The other classifiers included in our experiments do not hold hyperparameters for optimization. For the GMM based methods, the chord-scale models are trained on the same development set as in the SVM classification procedure and tested on the test split. Similarly, 10 iterations on the pseudo randomized splits are performed and the average is reported. Since the BTM method does not require training, the classifier is directly tested on the same test sets.

Table 2 shows the classification performances of the algorithms presented in this paper alongside the comparison of *HPCP.mean* and *HPCP.std* features. The results are provided in terms of accuracy scores (%). *HPCP.std* features outperforms *HPCP.mean* for almost all methods

which implies *standard deviation* is a more suitable choice of statistical summarization of features for this task.

For MLE based classification, additive method performs better than multiplicative likelihood estimation method. This may be due to how the features are processed, as mentioned in Section 4.3. The unfiltered learned GMM models perform worse than predefined binary template matching (BTM) algorithm. More so, multiplicative MLE performs poorly ( around 20% ). It appears that GMM models learn parameters that assign more weight on the out-scale pitch classes in the overall pitch class distribution of chord-scale models. These weights on out-scale pitch classes cause an expected decrease in accuracy score for MLE based classification as the method relies on the weights assigned to each pitch class by the mixture model.

Table 2: Results - Accuracy Scores

Method	HPCP.mean (%)	HPCP.std (%)
BTM-MLE (Mult.)	72.03	68.64
BTM-MLE (Add.)	79.23	79.66
SVM	76.25	80.83
GMM-MLE (Mult.)	19.92	21.18
GMM-MLE (Add.)	69.92	79.23
GMM-MLE-filt (Mult.)	71.19	71.61
GMM-MLE-filt (Add.)	76.69	<b>84.32</b>

The accuracy scores increase evidently when MLE is applied on GMMs that are filtered using the predefined binary chord-scale templates. These filtered GMMs show a performance increase between 6 – 60% depending on the chord-scale recognition method and the feature set used for classification. Leveraging domain knowledge appears to be effective for our task. Overall, *Additive MLE on filtered GMMs using HPCP.std features* perform the best among all the proposed algorithms. For comparison with the baseline, aforementioned algorithm outperforms the method proposed in [7] applied in this context by around 15%.

SVMs have comparable performance with filtered GMM-MLE. Note that using *HPCP.std* features perform better than using the *HPCP.mean* feature, supporting our claim that *std* features are a better fit as features for the classification tasks included in this study.

In Figure 3, the confusion matrix of the best performing algorithm and the feature set is provided. The chord-scale type with highest error rate is the *minor* and *altered* chord-scale types. The misclassified instances for minor scale are classified as either *phrygian*, which differs from minor scale by only one pitch-class. Most misclassified instances for *altered* scale are estimated as *half-whole diminished* scale. Even though these scales have different functionalities in tonal harmony, they are similar to each other in terms of diatonic pitch classes or scale degrees, which seem to be the major source error in classification. To overcome this problem, tonal harmonic constraints may be further applied to increase the classification performance. In general, this algorithm shows a reasonable performance and makes musically reasonable mistakes.

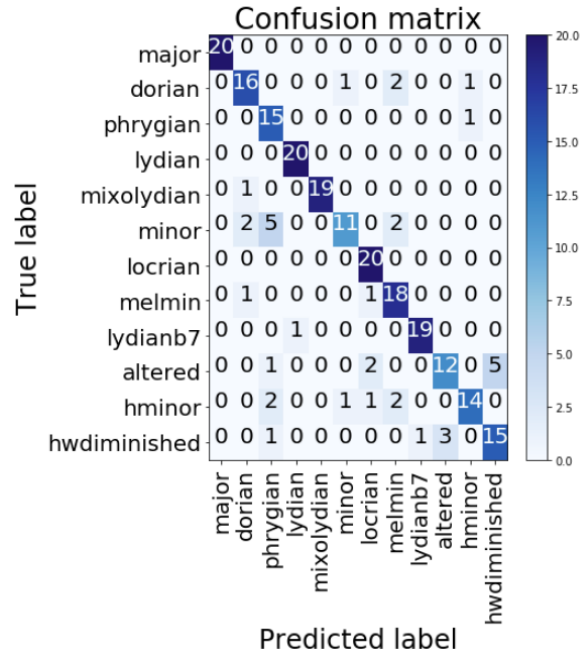


Figure 3: Confusion Matrix for the best performing method (GMM-MLE (filtered) with HPCP.std features)

## 9. CONCLUSION

We have tackled a relatively new domain of analysis in Music Information Retrieval research, which focuses on the retrieval of chord-scale information from improvised Jazz solos. The automatic recognition of chord-scale would be of use for various musical applications including style recognition or transfer, automatic harmonic analysis, performer identification and educational purposes. Our study applies a comparative study between a rule-based and two conventional machine learning approaches. The proposed methods were also used in a task specific manner for other related MIR tasks like chord recognition [13] [8], key detection [23] [24] [16]. The results of our study show a similar trend with the prior study on these related tasks. GMMs alone do not perform any better than the predefined binary template (BT) matching method. However, filtering the GMM scale models with these predefined templates is shown to be beneficial for the proposed chord-scale recognition pipeline. This result reveals the advantage of incorporating prior domain knowledge with learned models. Moreover, summarization of frame-based HPCP features over phrase-wise audio segments for chord-scale classification using their standard deviations perform more robustly compared to summarizing in terms of the mean of frame-based features, which agrees with the prior study in [20]. We have conducted the experiments on the Chord-Scale dataset, which is introduced in this work and shared publicly for reproducible and open science. The code to reproduce the experiment results and scale annotations can be found in the github repository. There are numerous possible future directions that the study presented here can take. Chord-scale recognition can be performed on MIDI notes (features) transcribed with an advanced AMT algo-



rithm. For skipping the automatic transcription step, there are several chroma extraction methods that provide features that are more robust to noise or the variance of the instrument type [13] [25]. The application of more advanced machine learning methods like neural networks require big-scale datasets. Hence, new strategies for obtaining more data in the context of Jazz improvisation need to be explored.

### Acknowledgments

The authors acknowledge Toprak Barut and Hikmet Altunbaslier for all their help in data collection. This work is supported by ERC funded TECSOME project. The author E.D received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 765068.

### 10. REFERENCES

- [1] X. Serra, "The computational study of a musical culture through its digital traces," in *Acta Musicologica*, vol. 89, no. 1. International Musicological Society, 2017, pp. 24–44.
- [2] B. Nettles and R. Graf, *The Chord Scale Theory and Jazz Harmony*, 1997.
- [3] W. A. Fraser, "Jazzology: A study of the tradition in which jazz musicians learn to improvise (afro-american)," in *9th Conference on Interdisciplinary Musicology*, 2014, 1983. [Online]. Available: <https://repository.upenn.edu/dissertations/AAI8406667>
- [4] D. Baker, *Jazz Improvisation (Revised): A Comprehensive Method for All Musicians*. Alfred music, 2005.
- [5] J. Aebersold and P. A. LONG, *Volume 1 [-] of A new approach to jazz improvisation*. J. Aebersold, 1974.
- [6] M. Levine, *The Jazz Theory Book*. O'Reilly Media, Inc., 2011.
- [7] C. Weiss and J. Habryka, "Chroma based scale matching for audio tonality analysis," in *Proceedings of 9th Conference on Interdisciplinary Musicology*, 2014, pp. pp.168–173. [Online]. Available: <http://publica.fraunhofer.de/dokumente/N-345665.html>
- [8] T. Cho, R. J. Weiss, and J. P. Bello, "Exploring common variations in state of the art chord recognition systems," in *In proceedings of the Sound and Music Computing Conference*, 2010, pp. 1–8.
- [9] N. Jiang, P. Grosche, V. Konz, and M. Müller, "Analyzing chroma feature types for automated chord recognition," in *Audio Engineering Society Conference: 42nd International Conference: Semantic Audio*, 2011.
- [10] V. Eremenko, E. Demirel, B. Bozkurt, and X. Serra, "Audio-aligned jazz harmony dataset for automatic chord transcription and corpus-based research," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pp. 483–490. [Online]. Available: <https://doi.org/10.5281/zenodo.1291834>
- [11] T. Fujishima, "Real-time chord recognition of musical sound: A system using common lisp music," in *Proceedings of International Computer Music Conference*, 1999, pp. pp.464–467. [Online]. Available: <http://hdl.handle.net/2027/spo.bbp2372.1999.446>
- [12] G. H. Wakefield, "Mathematical representation of joint time-chroma distributions," in *Proceedings of Advanced Signal Processing Algorithms, Architectures, and Implementations IX*, vol. 3807, 1999, pp. 637–646.
- [13] M. Muller and S. Ewert, "Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features," in *Proceedings of International Society of Music Information Retrieval Conference*, 2011, pp. pp.215 – 220. [Online]. Available: <http://doi.org/10.1.1.399.9397>
- [14] E. Gómez, "Tonal description of music audio signals," Ph.D. dissertation, Universitat Pompeu Fabra, 2006. [Online]. Available: <https://doi.org/10.1287/ijoc.1040.0126>
- [15] D. Bogdanov, N. Wack, E. Gómez Gutiérrez, S. Gulati, P. Herrera Boyer, O. Mayor, G. Roma Trepas, J. Salamon, J. R. Zapata González, and X. Serra, "Essentia: An audio analysis library for music information retrieval," in *In proceedings of International Society of Music Information Retrieval Conference*. [Online]. Available: <http://hdl.handle.net/10230/32252>
- [16] E. Gómez, "Key estimation from polyphonic audio," in *Music Information Retrieval Evaluation Exchange (MIREX05)*, 2005.
- [17] J. Serra, E. Gómez, P. Herrera, and X. Serra, "Chroma binary similarity and local alignment applied to cover song identification," in *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 6. IEEE, 2008, pp. 1138–1151. [Online]. Available: <https://doi.org/10.1109/TASL.2008.924595>
- [18] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152. [Online]. Available: <http://doi.org/10.1145/130385.130401>
- [19] V. N. Vapnik, "An overview of statistical learning theory," in *IEEE transactions on Neural Networks*, vol. 10, no. 5. IEEE, 1999, pp. 988–999. [Online]. Available: <http://doi.org/10.1109/72.788640>
- [20] E. Demirel, B. Bozkurt, and X. Serra, "Automatic makam recognition using chroma features," in *Proceedings of Folk Music Analysis Conference*, 2018. [Online]. Available: <https://doi.org/10.5281/zenodo.1239435>

- [21] J. H. Jensen, D. P. Ellis, M. G. Christensen, and S. H. Jensen, "Evaluation of distance measures between gaussian mixture models of mfccs." in *Proceedings of International Society of Music Information Retrieval Conference*, 2007, pp. 107–108. [Online]. Available: <https://www.ee.columbia.edu/~dpwe/pubs/JenECJ07-gmmdist.pdf>
- [22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," vol. 12, pp. 2825–2830, 2011. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1953048.2078195>
- [23] Ö. Izmirli, "Template based key finding from audio." in *International Computer Music Conference*, 2005. [Online]. Available: <http://www.music.mcgill.ca/~ich/research/misc/papers/cr1293.pdf>
- [24] G. Peeters, "Chroma-based estimation of musical key from audio-signal analysis," in *In proceedings of International Society of Music Information Retrieval Conference*, 2006, pp. 115–120.
- [25] M. Mauch and S. Dixon, "Approximate note transcription for the improved identification of difficult chords." in *IProceedings of International Society of Music Information Retrieval Conference*, 2010, pp. 135–140.

# Interacting with digital resonators by acoustic excitation

Max Neupert

Bauhaus-Universität Weimar  
max.neupert@uni-weimar.de

Clemens Wegener

The Center for Haptic  
Audio Interaction Research  
clemens@chair.audio

## ABSTRACT

This demo presents an acoustic interface<sup>1</sup> which allows to directly excite digital resonators (digital waveguides, lumped models, modal synthesis and sample convolution). Parameters are simultaneously controlled by the touch position on the same surface. The experience is an intimate and intuitive interaction with sound for percussive and melodic play.

## 1. INTRODUCTION

Motivation for the development of the instrument *Tickle* was a more intimate [2] and musical interaction with digital waveguides, lumped models, modal synthesis, sample convolution, as well as feedback-delay lines and filters. Aforementioned synthesis models can be subsumed as digital resonators. The instrument and questions about its driver architecture are discussed in [3]<sup>2</sup>.

## 2. THE TICKLE INSTRUMENT

### 2.1 Excitation, Material and Texture

To create an acoustic excitation signal we rely on a hard material that captures the spectra of different gestures. In addition to the rigidity of the material, a textured surface is essential to create enough noise when rubbed and wiped. Silicone surfaces are not suitable for our application since they absorb too much of the subtle interaction. A hard surface allows different spectra to propagate towards the piezoelectric sensor, creating vastly different responses in the digital resonators whether it is hit by a thumb, nail, ring or bowed with a violin bow on its edge. Percussive gestures like hits, knocks, flicks and continuous interactions like rubbing, scratching, or bowing can equally be captured.

### 2.2 Residual and Resonance

We want the interface to resonate as little as possible, so that we can feed this dry residual signal of the touch ges-

<sup>1</sup> In the literature the term *hybrid controller* [1] is found

<sup>2</sup> See this earlier publication for further references to related literature as they can't be included in this two-page demo paper

ture as excitation signal into a digital resonator (See also [4]). This way the full power of physical modeling synthesis algorithms may be accessed. The practice of sending generated noise-bursts or clicks into digital resonators which can be found in literature for physical modeling and which is still the standard in many soft- and hardware implementations is crippling the true potential of such algorithms.

### 2.3 Synthesis

Our synthesis algorithms are implemented as Pure Data patches and are available through our Git repository.<sup>3</sup>

For the sound synthesis we employ techniques of digital reverbrators. They can be understood as modeled simulations (waveguides and mass-spring models) of the physics happening in real instruments as described by Smith [5]. These models can be generated with Berdahl and Smith's Synth-A-Modeler Compiler [6]. Synth-A-Modeler generates FAUST code which can be compiled in a variety of other formats such as a Pure Data external. With the Pure Data object `pmpd~` from Henry's PMPD [7] library which can create static mass and spring models we achieved nice sounding string, plate and gong topologies. Drawback of PMPD is that the topographies and properties of the model can't be interactively modified while sound is processed.

We are not aiming for perfect recreations of orchestral instruments, our interest lies in the exploration of synthetic sounds with an acoustic and intimate level of control. Algorithms like a nested comb filter delay as described by Ahn and Dudas [8] prove interesting and fun to interpret with our instrument while being surprisingly cheap to compute. We can employ our acoustic interface to excite *extended*, *hybrid* and *abstract cyberinstruments* as described by Kojs et al. [9]. Convolution methods with samples can be useful to digital Foley artists to articulate a sample in a plenitude of variations.

### 2.4 Gestural Augmentation

To augment the excitation signal from the piezoelectric contact microphone, we gather the X and Y position of the touch event. A touch event lasts from the beginning of a touch until the release. We may also refer to it as a gesture. It can be translated to a note on and off with the note depending on in which hexagon the touch happened. While the gesture is lasting we can derive the X and the Y **offset** from the beginning of the touch event to the current

<sup>3</sup> [gitlab.chair.audio mirror: github.com/chairaudio](https://gitlab.chair.audio.mpg.de/chairaudio)

touch position. It is a useful modulation parameter for the synthesis. Naturally this coordinate offset can also be expressed in triangulated *distance* and *angle* between current position and origin of the gesture. Another useful modulation parameter is the speed of the movement.

When moving across the surface of the instrument, it may either be desired to trigger all notes like chimes or only allow the first note (or pitch) to be activated and thus allow for larger gestures extending to the whole surface while still playing the initial note.



Figure 1. *The Tickle instrument*

### 3. CONCLUSIONS

We believe only a hands-on experience with our instrument can convey the qualitative leap in intuitive control and intimate interaction with a musical instrument.

Testers reported that being able to discern a touch by the finger tip and the nail alone brings the interaction to a new level, that is new to melodic digital interfaces. The spectral and overall loudness response feels very natural and can be compared to that of an acoustic instrument.

Even though our instrument *Tickle* combines several well-known technologies which on their own may not be notable, in their combination they synergize to a powerful intuitive instrument which allows for a natural and intimate interaction with precise and reproducible control over sound. The existing technologies are touch pad, contact-microphone and physical modeling synthesis.

Feeding an analogue excitation signal into a (digital) resonator can create familiar as well as alien sounds. Sounds which either behave like instruments we know: Violin, guitar, snare drum, cymbal, gong, marimba, etc. or sounds which are distinctly synthetic but have an analogue touch to it. In a post-digital environment where “the paradigms analogue and the digital [...] exist simultaneously” [10, P.13] we believe that many new instruments will be seen in this new category of acoustic excitation instruments with digital resonators.

### 4. REFERENCES

- [1] R. Michon and J. O. Smith III, “A Hybrid Guitar Physical Model Controller: The BladeAxe,” in *Proceedings of the 2014 International Computer Music Conference*, A. Georgaki and G. Kouroupetroglou, Eds. Athens: International Computer Music Association, 2014, p. 7.
- [2] D. Wessel and M. Wright, “Problems and Prospects for Intimate Musical Control of Computers,” *Computer Music Journal*, vol. 26, no. 3, pp. 11–22, 2002.
- [3] M. Neupert and C. Wegener, “Isochronous Control + Audio Streams for Acoustic Interfaces,” in *Proceedings of the 17th Linux Audio Conference (LAC-19)*, CCRMA, Stanford University, Mar. 2019, p. 5. [Online]. Available: <http://lac.linuxaudio.org/2019/doc/neupert.pdf>
- [4] M. Puckette, “Infuriating Nonlinear Reverberator,” in *Proceedings of the 2011 International Computer Music Conference*. Huddersfield: International Computer Music Association, 2011, p. 4.
- [5] J. O. Smith III, “Physical Modeling Synthesis Update,” *Computer Music Journal*, vol. 20, no. 2, p. 44, 1996.
- [6] E. Berdahl and J. O. Smith III, “An Introduction to the Synth-A-Modeler Compiler: Modular and Open-Source Sound Synthesis using Physical Models,” in *Proceedings of the 2012 Linux Audio Conference*. Stanford: CCRMA, Stanford University, Apr. 2012, pp. 93–99.
- [7] C. Henry, “Physical modeling for pure data (PMPD) and real time interaction with an audio synthesis,” in *Proceedings of Sound and Music Computing 2004*, Paris, Oct. 2004.
- [8] J.-H. Ahn and R. Dudas, “Musical Applications of Nested Comb Filters for Inharmonic Resonator Effects,” in *Proceedings of the 2013 International Computer Music Conference*, vol. 2013. Perth: International Computer Music Association, 2013, pp. 226–231.
- [9] J. Kojs, S. Serafin, and C. Chafe, “Cyberinstruments via Physical Modeling Synthesis: Compositional Applications,” *Leonardo Music Journal*, vol. 17, pp. 61–66, Dec. 2007. [Online]. Available: <http://www.mitpressjournals.org/doi/10.1162/lmj.2007.17.61>
- [10] C. Thorén, M. Edenius, J. E. Lundström, and A. Kitzmann, “The hipster’s dilemma: What is analogue or digital in the post-digital society?” *Convergence: The International Journal of Research into New Media Technologies*, vol. 20, no. 10, p. 16, Jun. 2017. [Online]. Available: <http://journals.sagepub.com/doi/10.1177/1354856517713139>



# MELODY SLOT MACHINE

Masatoshi Hamanaka

RIKEN

masatoshi.hamanaka@riken.jp

## ABSTRACT

This paper describes our interactive music system called the “Melody Slot Machine,” which enables control of a holographic performer. Although many interactive music systems have been proposed, manipulating performances in real time is difficult for musical novices because melody manipulation requires expert knowledge. Therefore, we developed the Melody Slot Machine to provide an experience of manipulating melodies by enabling users to freely switch between two original melodies and morphing melodies.

## 1. INTRODUCTION

Our Melody Slot Machine provides a unique experience, enabling the control of a virtual performer. The Melody Slot Machine has these three features.

**User-Friendly Interface:** To enable anyone to easily control his or her virtual performer, we used a dial-type interface that enables replacing a part of the melody segment that the virtual performer will play (Figure 1a, 1b). The score is sandwiched between an acrylic board and a tablet. The dial interface on the tablet can be operated with fingers through a rectangular hole in the acrylic board. When the red lever on the right side of the score is pulled down, all the dials rotate, and one of the melody segments on the dial is randomly selected. Variations in melody segments are composed on the basis of the melody morphing method, so switching the melody segments maintains the overall structure of the melody and only changes the ornamentation [1].

**Easy-to-understand Control Results:** We prepared a display showing a performer so that the results of the control can be confirmed visually as well as aurally. A holographic display is used to show the performer so as to increase the feeling of presence (Figure 1c). The users can feel like they are controlling a performer by operating a melody.

**Improving the Feeling of Presence:** We recorded all the performance sounds to increase the feeling of presence. For the recording, we used a studio with very little reverberation; only the reverberation of the preceding sound enters the beginning of the melody segment because of the melody splitting into segments. Reverberation is added when the melody is played. Three pairs of speakers were installed, and the pan pot and reverb were set for each direction so that the hologram seemed to be a real performer (Figure 1d). By placing one’s head between two pairs of speakers, both the sound and video enhance the feeling of presence.

Copyright: © 2019 Masatoshi Hamanaka. This is an open-access article distributed under the terms of the [Creative Commons Attribution License 3.0 Unported](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 2. MELODY MORPHING METHOD

The Melody Slot Machine is a system that enables novices to experiment with melody manipulation on the basis of the melody morphing method [1].

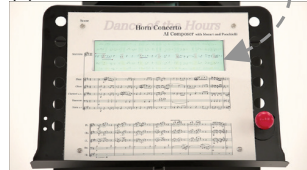
Figure 2 shows an example of abstracting a melody by using a time-span tree. There is a time-span tree from melody D, which embodies the results of the Generative Theory of Tonal Music (GTTM) analyses. The structurally important notes are connected to a branch close to the root of the tree. In contrast, ornamentation notes are connected to the leaves of the tree. We can obtain an abstracted melody E by slicing the tree in the middle and omitting notes that are connected to branches under line E.

In melody morphing, we use the primitive operations of subsumption relation (written as  $\sqsubseteq$ ), meet (written as  $\sqcap$ ), and join (written as  $\sqcup$ ), as proposed by Hirata [2]. Subsumption represents the relation by which “an instantiated object”  $\sqsubseteq$  “an abstract object.” The meet operator extracts the largest common part or the most common information of the time-span trees of two melodies in a top-down manner. The join operator joins two time-span trees in the top-down manner as long as the structures of two time-span trees are consistent.

(a) Slot dials

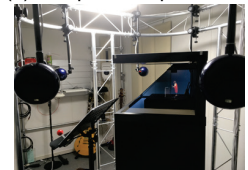


(b) Slot lever



Front side view

(d) Tree pairs of speakers



(c) Holographic display

Right side view



Figure 1. Melody Slot Machine components

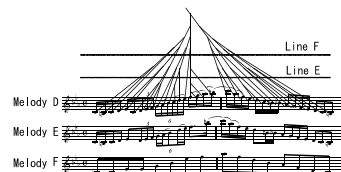


Figure 2. Abstraction of melody

By using the time-span trees  $T_A$  and  $T_B$  from melodies  $A$  and  $B$ , respectively, we can calculate the most common information,  $T_A \cap T_B$ , which are the essential parts of melody  $A$ , as well as those of melody  $B$ . The meet operations  $T_A \cap T_B$  are abstracted from  $T_A$  and  $T_B$ , and those discarded notes are regarded as the difference information of  $T_A$  and  $T_B$  (Figure 3a). We consider that there are features without the other melody in the difference information of  $T_A$  and  $T_B$ . Therefore, we need a method for smoothly increasing or decreasing these features. The melody divisional reduction method can abstract the notes of the melody in the differential branch of the time-span tree (Figure 3b). We use the join operator to combine melodies  $C$  and  $D$ , which are the results of the divisional reduction or augmentation using the time-span tree of melodies  $A$  and  $B$  (Figure 3c).

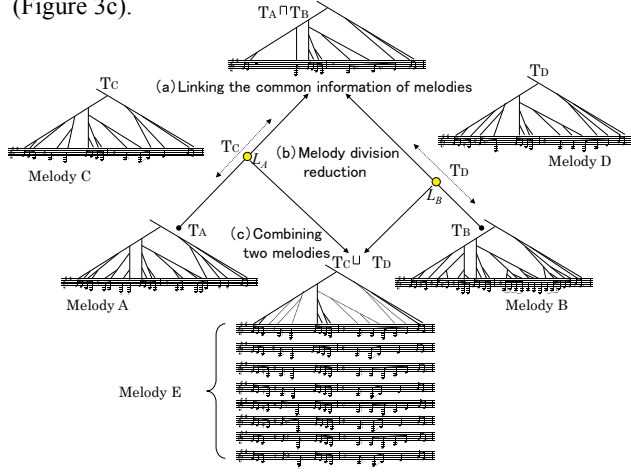


Figure 3. Melody morphing method

### 3. IMPLEMENTATION

The holographic display (Dremoc HD3) can be viewed from three directions by using three glass panes with semitransparent film and reflecting the display installed on the top of the device. Figure 4 shows the hardware implementation of the Melody Slot Machine.

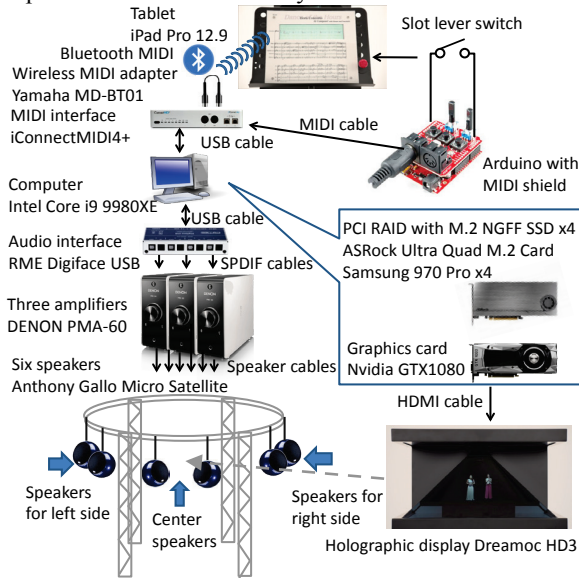


Figure 4. Implementation

The video signal is connected to the holographic display with an HDMI cable via the graphics card. The video data have been taken in advance from three directions for all melody tracks. When the user sets the dial numbers on the tablet, the system plays in accordance with those numbers. The video is large in size compared to that of the audio, and it takes more time to start playing the file, making it more complicated than sound processing.

First, all the video files are concatenated into one file, and two copies of that file are written to a disk as file 1 and file 2. Then, during the playback of video A in file 1, if the next video to be played back is B, file 2 seeks the playback position of video B, and playback occurs immediately. At the moment video A ends, it releases the connection of the renderer connected to file 1, connects to file 2, and plays video B. The switching of this renderer ends within one frame, which is within 33.3 milliseconds, so a smooth connection without dropped frames is apparent.

We implemented this algorithm by using the VIDDULL video engine in MAX/MSP and Apple ProRes 422 video co-dec with a frame rate of 30 fps.

### 4. EXPERIMENTAL RESULTS

With VIDDULL, we can set the cache size of the buffer to be read ahead when seeking the play back position. Table 1 shows the frame rate of the video when cache size changes. We used a cache size of 0.5 gigabyte, where the frame rate did not decrease much. The seeking of the playback position starts 0.5 seconds before playback. This is because with a cache size of 0.5 gigabytes, the time to seek is within 0.5 seconds. If the seeking time is less than 0.5 seconds, frame dropping occurs.

Table 1. Maximum and minimum frame rates

Cache size	Maximum frame rate	Minimum frame rate
0.01 gigabyte	27 fps	24 fps
0.05 gigabyte	27 fps	24 fps
0.08 gigabyte	28 fps	24 fps
0.15 gigabyte	28 fps	25 fps
0.50 gigabyte	30 fps	28 fps

### 5. CONCLUSION

In this paper, we described the Melody Slot Machine, which enables control of virtual performers on a holographic display. We plan to create various contents for the Melody Slot Machine in future works.

### 6. REFERENCES

- [1] M. Hamanaka, K. Hirata, and S. Tojo, "Melody Morphing Method Based on GTTM," *ICMC2008*, pp. 155–158, 2008.
- [2] K. Hirata and T. Aoyagi, "Computational Music Representation Based on the Generative Theory of Tonal Music and the Deductive Object-Oriented Database," *CMJ*, Vol. 27, No. 3, pp. 73–89, 2003.

# OM-AI: A TOOLKIT TO SUPPORT AI-BASED COMPUTER-ASSISTED COMPOSITION WORKFLOWS IN OPENMUSIC

**Anders Vinjar**  
Composer Researcher  
Oslo, Norway  
anders@avinjar.no

**Jean Bresson**  
STMS lab: IRCAM – CNRS – Sorbonne Université  
Paris, France  
jean.bresson@ircam.fr

## ABSTRACT

We present ongoing works exploring the use of artificial intelligence and machine learning in computer-assisted music composition. The OM-AI library for OpenMusic implements well-known techniques for data classification and prediction, in order to integrate them in composition workflows. We give examples using simple musical structures, highlighting possible extensions and applications.

## 1. INTRODUCTION

The idea of making programs capable of composing appeared early in the history of computer music [1]. Today artificial intelligence and machine learning are commonly used for research on computational creativity [2], “autonomous” generative and/or improvisation systems [3–5], or real-time performance monitoring and interaction [6]. However, apart from a few examples [7, 8], machine learning and AI are rarely exploited by composers as a means for writing music.

Computer-assisted composition systems develop explicit computational approaches through the use of end-user programming languages [9]. At the forefront of this approach, OpenMusic is a popular visual programming environment allowing users to process and generate scores, sounds and many other kinds of musical structures [10].

We present ongoing works exploring the use of AI and machine learning techniques in this environment. In contrast to approaches aimed at autonomous creative systems, our aim is to apply these techniques as composition assistance in the classification and processing of musical structures and parameters. Therefore we target a “composer-centered” machine learning approach [11] allowing users of computer-assisted composition systems to implement experimental cycles including preprocessing, training, and setting the parameters of machine learning models for the data generation, decision support or solving other generic problems.

Copyright: © 2019 Anders Vinjar and Jean Bresson. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 2. TOOLS AND ALGORITHMS

The OM-AI library for OpenMusic provides basic tools from the domain, with elementary algorithms to classify vectors in a multidimensional feature space [12].

### 2.1 Vector Space

A generic data structure called VECTOR-SPACE is used to store vectorized data and information necessary to train and run machine learning and classification models. The structure is simple and generic; it is initialized with a list of entries (key, value) for a hash-table of vectors, where keys can be strings or any other unique identifiers for the different vectors.

Feature-vectors are also stored as hash-tables using descriptor names as keys. It is assumed that each feature-vector contains the same set of descriptors. Descriptor names can also be input to the VECTOR-SPACE initialization for facilitating visualization and query operations. A graphical interface allows the 2D and 3D visualization of vectors in the feature space, selecting two or three descriptors as projection axes (see Figure 1).

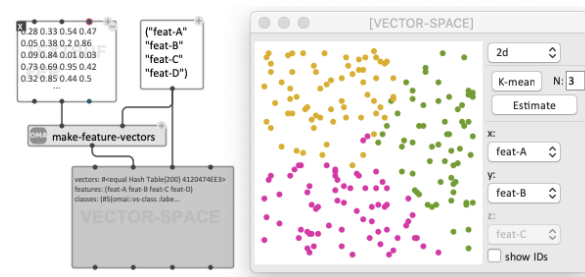


Figure 1. 2-D vector space visualization.

### 2.2 Clustering and Classification

Within the VECTOR-SPACE, built-in or user-defined distance-functions are used to compute measurements of similarity between feature-vectors (i.e. distance within the feature space) and centroids for any set of vectors. These operations are applied in various algorithms for automatic clustering and classification.

The *k-means* algorithm performs “unsupervised” clustering by grouping feature-vectors around a given number of centroids. This process can be done in a visual program (see Figure 2) or interactively from within the VECTOR-SPACE graphical interface (as in Figure 1).

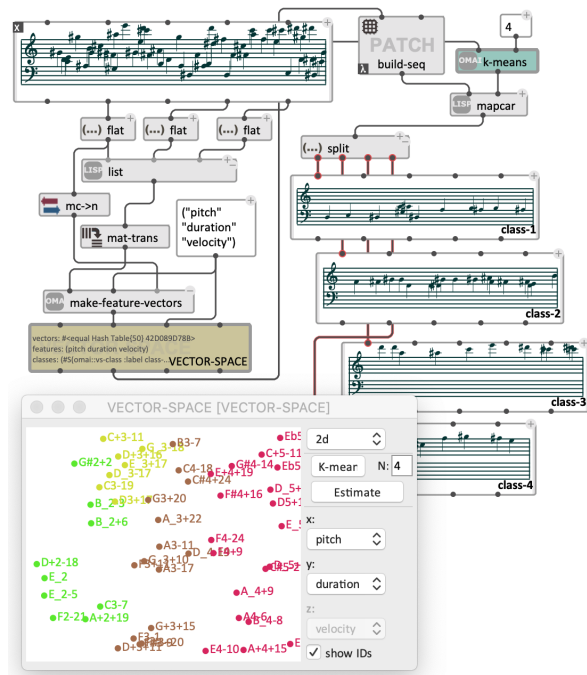


Figure 2. A simple example of clustering applied to a sequence of notes. Feature-vectors are formatted and input to the VECTOR-SPACE object. From there, the *k-means* algorithm clusters the data based on the specified features (*pitch*, *duration*, *velocity*). Vectors are output and split by clusters to generate four separate voices.

Supervised classification approaches (based on preliminary labelling information) are also available. In our generic model, a *class* is represented by a unique label and a list of IDs corresponding to known members of this class (this is typically determined during a preliminary training stage). Based on this information (which implicitly labels all known class members), it is possible to compare any unlabelled vector with centroid feature-vectors of the different classes, or its similarity with an established neighbourhood in the multidimensional feature space (*k-NN*). Such comparison allow to determine a measure of likelihood for this vector belonging to a certain class.

### 3. MUSICAL DESCRIPTORS

An extensible set of description algorithms allows to extract features from musical objects (harmonies, chord sequences, temporal attributes etc.). These features can be combined freely to constitute the vectors representing musical data in the different OM-AI algorithms.

Currently available descriptors include pitch statistics: most common pitch or pitch class, pitch histograms, mean pitch, pitch variety, interval between prevalent pitches, relative pitch prevalence, importance of different registers, tonal centers, etc., and melodic features: intervals, arpeggios and repetitions, melodic motions and arcs, etc.

We are currently working on extending the set of provided feature extraction algorithms and experimenting with more advanced musical examples.

### 4. RESOURCES AND DOWNLOAD

The sources of OM-AI are open and available along with documentation and examples at:

<https://github.com/openmusic-project/OMAI>

### Acknowledgments

This work is supported by the PEPS 3IA programme of the French National Center for Scientific Research, IRCAM's Artistic Research Residency Program and Notam.

### 5. REFERENCES

- [1] L. A. Hiller and L. M. Isaacson, *Experimental Music: Composition With an Electronic Computer*. McGraw-Hill, 1959.
- [2] P. Pasquier, A. Eigenfeldt, O. Bown, and S. Dubnov, "An Introduction to Musical Metacreation," *Computers in Entertainment*, vol. 14, no. 2, 2016.
- [3] F. Ghedini, F. Pachet, and P. Roy, "Creating Music and Texts with Flow Machines," in *Multidisciplinary Contributions to the Science of Creative Thinking*, G. E. Corazza and S. Agnoli, Eds. Springer, 2015.
- [4] G. Assayag, S. Dubnov, and O. Delerue, "Guessing the Composer's Mind: Applying Universal Prediction to Musical Style," in *Proc. International Computer Music Conference (ICMC'99)*, Beijing, China, 1999.
- [5] L. Crestel and P. Esling, "Live Orchestral Piano, a system for real-time orchestral music generation," in *Proc. Sound and Music Computing (SMC'17)*, Espoo, 2017.
- [6] J. Françoise, N. Schnell, and F. Bevilacqua, "A Multimodal Probabilistic Model for Gesture-based Control of Sound Synthesis," in *Proc. ACM MultiMedia (MM'13)*, Barcelona, 2013.
- [7] D. Cope, *Experiments in Musical Intelligence*. A-R Editions, 1996.
- [8] S. Dubnov and G. Surges, "Delegating Creativity: Use of Musical Algorithms in Machine Listening and Composition," in *Digital Da Vinci: Computers in Music*, N. Lee, Ed. Springer, 2014.
- [9] G. Assayag, *et al.*, "Computer Assisted Composition at IRCAM: From PatchWork to OpenMusic," *Computer Music Journal*, vol. 23, no. 3, 1999.
- [10] J. Bresson, C. Agon, and G. Assayag, "OpenMusic. Visual Programming Environment for Music Composition, Analysis and Research," in *Proc. ACM MultiMedia (MM'11)*, Scottsdale, 2011.
- [11] M. Gillies, R. Fiebrink, A. Tanaka, *et al.* "Human-Centered Machine Learning Workshop at CHI'16," in *Proc. CHI'16 – Extended Abstracts on Human Factors in Computing Systems*, San Jose, 2016.
- [12] C. D. Manning, P. Raghavan, and H. Schütze, *An Introduction to Information Retrieval*. Cambridge University Press, 2009.



# URALI: A PROPOSAL OF APPROACH TO REAL-TIME AUDIO SYNTHESIS IN UNITY

Enrico Dorigatti

Conservatorio di Musica "F. A. Bonporti"  
enricodorigatti@rocketmail.com

## ABSTRACT

This paper aims to give a basic overview about the URALi (Unity Real-time Audio Library) project, that is currently under development. URALi is a library that aims to provide a collection of software tools to realize real-time sound synthesis in applications and softwares developed with Unity.

## 1. INTRODUCTION

Unity is a developing environment used for the creation of software applications, in particular videogames and mobile apps. Today is well recognized due to the amount of instruments it offers within its environment, as well as for the great quality of the final product in general, and of the graphic in particular. It has, also, a huge community of users all over the world, made of people helping each other in a collaborative atmosphere.

Unity is well known for the visuals quality, also if everyone knows that an app -and also more a game- needs not only high level graphic, but also a great audio to result very effective and addictive. Pre-made audio clips are very well supported within Unity, with a variety of effects to manipulate it, and a virtual mixer where to mix together various sources.

However, there is no only the case where scenes are built by the developer and there is a story to follow. In fact, there is also the case of the generative and algorithmic art<sup>1</sup>, today quite popular, as well as the sonification one, where there are little or no rules, and the application evolves autonomously during time, based on algorithms and random events, as well as on rules. For those cases, a developer will likely not use a premade clip, but maybe will search for something more organic and maybe uncommon, and, most of all, that can evolve during time related to the application's visuals.

Of course there are way to control synthesis-dedicated softwares like Supercollider and Max/MSP via OSC protocol, but going that way means the use of external tools that can work (interpret and send/receive) OSC data, and, more important, knowledge about how to operate and

program an audio synthesis dedicated environment like the ones mentioned above.

Natively, inside Unity, there are no instruments that can produce sound and that can be driven by an algorithm, instruments that are a must to carry out an evolving and unpredictable sonification; instead, there is a base where to start to build a sound synthesis chain, a very interesting and useful native function called `OnAudioFilterRead`. It is called from a separate thread and it aims to fill a buffer with samples that are going to represent sounds when played by the speakers. However, there are no classes providing objects with which easily synthesize basic waves or carry out other types of sound synthesis; plus, creating a synthesis chain within `OnAudioFilterRead` is not so immediate and can easily lead to confused code, as well as a loss of performance. From here, the decision to build some personal instruments in order to obtain an algorithmic sonification, ended in building from scratch a solid and optimized library containing instruments for the so called "academic" audio synthesis.

## 2. GENESIS

The first project of URALi was a simple sinusoidal oscillators class meant to carry out a precise and specific task, that was providing sinusoidal waves's data ready to use in various synthesis processes (ring modulation, additive and FM), to create the algorithmic, real-time and visual-related sonification of *Life*, an audiovisual generative software based on a custom version of the famous J. Conway's Life algorithm.

However, the project grew up very quickly, eventually becoming the base where to start the building of a bigger library to be used for the procedural and academic-like sonification of any Unity project using it, bringing also an added value that comes from the proven extreme simplicity to integrate and correlate, in this fully multimedial environment, audio data with visual outputs and/or algorithms, and vice-versa.

## 3. INSIDE URALI

URALi is, at the moment, a full-working audio library that offers basic objects for the audio synthesis. In this paragraph is proposed an overview of the actual state of completion of the library, as well as the main improvements scheduled to be implemented as soon as possible, since there is still a lot of work to do in order to make URALi a really powerful instrument.

Copyright: © 2019 Enrico Dorigatti. This is an open-access article distributed under the terms of the [Creative Commons Attribution License 3.0 Unported](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup>E. Dorigatti, "Life"; G. Albini, "Memoriale"

### 3.1 Actual state of completion

At the moment, URALi provides an implementation of the following object and features:

- sinusoidal, triangular, sawtooth and square oscillators;
- lookup oscillator;
- PolyBLEP algorithm to antialias harmonic waveforms;
- multi-segment (not limited to) ADSR envelope;
- granulator;
- white noise source;
- convenience objects (enveloped oscillator, additive module);
- multithreading.

### 3.2 Scheduled features

Among other minor features, and improvements, the priority ones are:

- wavetable to play user's own waveforms;
- GUI to tweak the parameters inside Unity editor and not only inside the code;
- extension of the convenience objects section, adding, among the others, modules for other types of synthesis;
- miscellaneous utility features, like the possibility to record to a wav file and to view the spectrogram and the waveform generated.

## 4. THE LOGIC OF URALI

It is worth to distinguish between how URALi works in the background and how is meant to be used on the final user side, to have at last an idea of the whole process.

### 4.1 Internal synthesis logic and design

The idea behind URALi is to avoid the saturation of the `OnAudioFilterRead` function with any kind of wave calculation or wavetable's sample interpolation. So it leaves all the calculation tasks to a separate and much more faster background thread, that eventually returns the result of all operations, allowing `OnAudioFilterRead` to carry out minor tasks, like amplitude or frequency control, without any performance loss, that would mean discontinuities in the waveform.

### 4.2 User side design

On the user side, URALi needs the user to write its own audio synthesis chain inside a function that has to be passed as the argument for another function that starts the audio engine. Then, various parameters -such as frequen-

cy or amplitude- of the objects can be modified in runtime thanks to public properties, changing the acoustic characteristics of the final output. To achieve an audible result, then, one has to recall inside of `OnAudioFilterRead` the function of the library that returns the samples, making possible to hear a sonic result at last.

## 5. CONCLUSIONS

URALi is still in development and that leads to having, right now, an instrument that provides only the very essential tools for the audio synthesis, also if those work perfectly. The hope is that the project (as well as the generative art as discipline) can interest more artists and Unity developers everyday, supported by a very powerful and friendly development environment.

Last note is that URALi, first of all, is my personal approach and solution to the lack of instruments for generative audio that I found in Unity. That means that it may not be the perfect solution, nor the only one. It works very well but, as the title says, it is a proposal.

# A SEQUENCER WITH DECOUPLED TRACK TIMING

Silvan David Peter, Gerhard Widmer

Institute of Computational Perception, Johannes Kepler University, Linz  
 {silvan.peter, gerhard.widmer}@jku.at

## ABSTRACT

Sequencers almost exclusively share the trait of a single master clock. Each track is laid out on an isochronously spaced sequence of beat positions. Vertically aligned positions are expected to be in synchrony as all tracks refer to the same clock. In this work we present an experimental implementation of a decoupled sequencer with different underlying clocks. Each track is sequenced by the peaks of a designated oscillator. These oscillators are connected in a network and influence each other's periodicities. A familiar grid-type graphical user interface is used to place notes on beat positions of each of the interdependent but asynchronous tracks. Each track clock can be looped and node points specify the synchronisation of multiple tracks by tying together specific beat positions. This setup enables simple global control of microtiming and polyrhythmic patterns.

## 1. INTRODUCTION

Sequencer user interfaces usually have a grid-like structure with musical time in the x-axis and the number of tracks in the y-axis. All values in the horizontal refer to the same discretisation of musical time, often sixteenth notes. As users place notes on this grid, they expect vertically aligned points to play in synchrony. This setup is virtually universal but also notoriously unintuitive for polyrhythmic and microtiming pattern generation. Modern sequencers offer several workarounds, such as the possibility to place notes on a much finer subgrid or the possibility to have repeating sequences of different length for each track. In this demonstration we decouple the tracks by using microtiming patterns generated by coupled oscillator networks as beat positions for each track individually.

## 2. THE OSCILLATOR NETWORK

A simple oscillator network is modelled as two connected ordinary differential equations with reset mechanism. The oscillators peak when their respective values reach a threshold level  $u_{peak}$ . A modifying oscillator  $u_2$  influences the

time derivative of a salient oscillator  $u_1$  at peaks:

$$\tau_1 \frac{du_1}{dt} = u_1 + c_{12} \delta(u_{peak} - u_2) \quad (1)$$

$$\tau_2 \frac{du_2}{dt} = u_2 \quad (2)$$

Each  $u_i$  is reset to  $u_{reset}$  when passing a threshold  $u_{peak}$ . We implement an expanded network of eight oscillators with modifiable interdependencies  $c_{ij}$ , initial values, and frequencies  $\tau_i$ . Each oscillator's peak times are recorded as sequence of anisochronous beat times.

## 3. THE SEQUENCER

The sequencer consists of eight tracks, each sequenced by an assignable oscillator. Each track consists of a definable number of beat positions visually arranged in the usual grid structure. The timing of the positions is computed by the sequence of peaks of its oscillator. A segment of the oscillator output can also be looped. To reduce the chaos of completely independently repeating patterns of different length, two looped tracks can be tied together at node points in the sequence of beat positions. Two tracks that share a node point will play the respective positions simultaneously. The track higher on the y-axis defines the absolute length of the loop, the other loop gets dilated or shrunk to fit. In the same way subsequences can be distorted by setting multiple node points. The sequencer will be made available as web-based tool for experimentation.

## 4. SUMMARY

By multiplying, connecting, and decoupling the underlying beat oscillations from the familiar grid interface we are able to implement a simple sequencer with expanded microtiming and polyrhythm capabilities. A few changes of the parameters generate for instance quintuplets over a wonky four on the floor and a continuously evolving completely distorted timing pattern on top. High-level control of the whole sequence offers a musically intuitive approach to timing pattern generation. It is our hope that this sequencer architecture facilitates experimentation where editing single notes becomes too tedious.

## Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 670035).

Copyright: © 2019 Silvan David Peter, Gerhard Widmer et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

# MUSICYPHER: MUSIC FOR MESSAGE ENCRYPTION

V. Jaime, A. Peinado

Escuela Técnica Superior de Ingeniería de Telecomunicación, Universidad de Málaga  
victorjm96@uma.es, apeinado@ic.uma.es

## ABSTRACT

An Android application has been developed to encrypt messages using musical notes that can be automatically played from the smartphone and/or stored in a midi file to be transmitted over any available connection. The app has been designed to recover the original message on-the-fly detecting the notes played by a different device. The main objective of this project is to make known the relationship between cryptography and music showing old systems (XVII century) implemented in modern devices, the smartphones, using the tools they provide us, such as the microphone, the speakers, and the internal storage.

## 1. INTRODUCTION

The encryption of information has always been a matter that the human being has been developing over the years [7], whether for war purposes or as in the present, for example, for privacy of sensitive data in the network [8]. One of the most used techniques since the first attempts to hide the information has been the substitution of the message's characters by other symbols, such as letters of the same or another language, numbers or symbols invented for the occasion.

One of the best treatises in Cryptography of all time, written by Lieutenant Carmona in 1894 [6], includes the cryptosystem proposed by Guyot, based on the idea of Gaspar Schott (1667), of using musical notes to encrypt messages. Although this is not the first cryptosystem using music as a vehicle to encrypt or hide the information, it can be considered as the most representative one.

Musicypher is an Android application that brings to the present this ancient cryptosystem with several objectives in mind: on the one hand, highlight the existing relationship between music and cryptography since many years; on the other hand, bring cryptography closer to the general public through the music, and conversely, promote the musical learning through games using (in this case) the message encryption.

*Copyright: © 2019 V. Jaime, A. Peinado. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.*

Section 2 describes the Guyot cryptosystem with more detail. Section 3 presents the more relevant aspects of the software developed. Finally, several conclusions are provided in section 4.

## 2. THE GUYOT CRYPTOSYSTEM

There have been various types of encryption, some quite original such as Guyot in the seventeenth century, which uses music as a way to hide the original text. Using an artifact, a musical note or chord was assigned to each letter and, with this same device, the sender and receiver of the message could encrypt and decipher the information.

Guyot's device consists of two concentric circles. In one of them the 26 letters of the alphabet are represented, and in the other the note or musical chord with which it will be encrypted. With this device, the notes would be written in a score. Of course, the final result had no musical sense, but it went unnoticed among people without musical knowledge. When the recipient receives the score, the message could be obtained with the same device that was used to encrypt the original.

## 3. THE ANDROID APPLICATION

Musicypher is an application that consists of the implementation of the Guyot crypto-system adapted to current technology using a very common device: the Smartphone. The main objective is this adaptation, but not the improvement of its security.

Musicypher is a program for Android systems, developed in Java using the Android Studio programming environment, in which the encryption of a text string is carried out transforming it in a sequence of musical notes. Similarly, the corresponding deciphering is implemented.

The transmission of encrypted messages can be carried out in two ways: by means of sound, by playing a coded music message in order to be captured by the microphone of a receiver smartphone, and by means of an audio MIDI [1] file, without the need for reproduction. Therefore, the decryption has been implemented such that the message is recovered from the MIDI file, or detected in real-time while the sound is being played by the transmitter device.



### 3.1 Encryption of the message

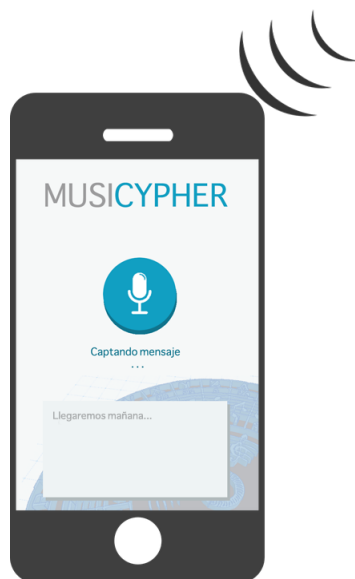
In order to provide more options to the final user, before the musical part he will be offered different types of encryption systems to be applied to the text, basically based on substitution or transposition, which require of a key that the receiver must also know in order to decode the information. After this part the text-to-music conversion takes place in which, as in the Guyot system, musical notes are assigned to the alphabet.

Once the encrypted file is generated, it can be stored to be later shared with other users, and played, within the same application.

### 3.2 Decryption of the message

The deciphering of a text can be done through the microphone of the device. It starts to catch the melody and each time a note change occurs, deciphers the fundamental frequency [4] [5] and concatenates the corresponding letter to the unique text string. The process will end when a note that means the end is recorded.

After that, the encrypted text message is obtained and through the substitution or transposition method and the source key it is decrypted, and the original message is obtained.



**Figure 1.** Real-time decoding of the musical message

### 3.3 Considerations

Text to music conversion is done using the MIDI standard, in which each musical note is identified by a number. Taking it into account, and assigning a duration time to each note, a ".MID" file is generated. This file can be reproduced by Musicypther or by any device that supports the standard.

In order to capture the sound, the technical limitations of mobile microphones [2] [3] have been taken into account, and we have chosen to use the frequency range of the

telephone channel band (300Hz to 3kHz), since this is where the frequency response is flatter in most devices and, therefore, the capture is more reliable.

## 4. CONCLUSIONS

The Guyot [6] encryption system has been adapted to an Android application, using the current technology available at usual smartphones, in order to deploy a complete communication system (transmission, encryption, decryption, reception). The current prototype is focused on the musical part. Further developments will take into consideration the improvement of the security part, allowing us to provide a more robust app.

### Acknowledgments

This work was partially supported by the Universidad de Málaga, Andalucía Tech, and by the Ministerio de Economía, Industria y Competitividad (MINECO), Agencia Estatal de Investigación (AEI), and Fondo Europeo de Desarrollo Regional (FEDER, UE) under Grant TIN2017-84844-C2-1-R (COPCIS project).

## 5. REFERENCES

- [1] J. Bartolomé. La especificación MIDI, 2013. URL: <http://tolaemon.com/docs/midi1.htm>
- [2] B.B. Morson. "Hearing voices in the high frequencies: What your cell phone isn't telling you", in Proc 168<sup>th</sup> Acoustic Society of America (ASA) Meeting, Indianapolis, October, 2014. Available at <https://acoustics.org/hearing-voices-in-the-high-frequencies-what-your-cell-phone-isnt-telling-you-brian-b-monson/>
- [3] Technical Direct. "Smartphone User Experience Analysis - Audio (II)", 2016. URL: <http://www.technical-direct.com/en/smartphone-user-experience-analysis-audio-ii/>
- [4] J. Six, O. Cornelis, M. Leman. TarsosDSP, a Real-Time Audio Processing Framework in JAVA. Univertisy College Ghent. 2014.
- [5] A. de Cheveigne, H. Kawahara. "YIN, a fundamental frequency estimator for speech and music", in J. Acoust. Soc. Am. 111 (4), 2002, pp. 1917-1930.
- [6] J. García Carmona. Tratado de criptografía con aplicación especial al ejército. Madrid: Sucesores de Rivadeneyra. 1894. Re-edited by Ministerio de Defensa, 2011.
- [7] D. Kahn, The codebreakers: the story of secret writing. New York: Macmillan. 1967
- [8] A.J. Menezes, P.C. Oorschot, S.A. Vanstone. Handbook of applied cryptography. Boca Raton, FL, USA: CRC Press, 5th printing. 2001

# A PLATFORM FOR PROCESSING SHEET MUSIC AND DEVELOPING MULTIMEDIA APPLICATION

**Fu-Hai Frank Wu**

National Tsing Hua University, Taiwan  
fhfwu@gapp.nthu.edu.tw

## ABSTRACT

Imaging when reading sheet music on computing devices, users could listen audio synchronizing with the sheet. To this end, the sheet music must be acquired, analyzed and transformed into digitized information of melody, rhythm, duration, chord, expressiveness and physical location of scores. As we know, the optical music recognition (OMR) is an appropriate technology to approach the purpose. However, the commercial OMR system of numbered music notation is not available as best as our knowledge. In the paper, we demonstrate our proprietary OMR system and show three human-interactive applications: sheet music browser and multimodal accompanying and games for sight-reading of sheet music. With the illustration, we hope to foster the usage and obtain the valuable opinions of the OMR system and the applications.

## 1. INTRODUCTION

There are commercial products of optical music recognition (OMR) for sheet music in staff notation, for example Sharpeye, Photoscore in Sibelius, Smartscore, and Midiscan in Finale. But, for the sheet music, as shown in Figure 1, of numbered music notation [1] called ‘jiǎnpǔ’ in pinyin for Mandarin, the functions of commercial product, it seems that the visibility of the OMR system is low. In our previous research [2], we design an eco-system of OMR for numbered music notation. The eco-system majorly includes the methods and utilities to generating groundtruth, recognition, and rendering of sheet music, and evaluation metrics of OMR system.

Such a notation has been used extensively in Asia for music practitioners to distribute their musical works because of well acceptance by the consumers. The notations include digits, alphabet, and characters as the same shape as ASCII symbols, and other glyphs such as ties, slurs, tuples, etc., as those of staff notation. An exemplar music sheet, traditional Chinese song, is shown in Figure 1.

Setting up the goal to improve the experience and ability of sheet music sight-reading, we develop the multimodal and multimedia applications. There are three major functions: sheet music browser, singing accompaniment and games for pitch and note length error detection.

## 2. OMR ECO-SYSTEM

Because the lack of public dataset, we build up the ecosystem for OMR system from the scratch. Figure 2 (a) shows the flow to construct the building blocks of the ecosystem. The system comprises of the four major processing blocks: 1) preprocessing of document image; 2) musical glyph definition and recognition; 3) music notation assembly by graphical and musical semantic; 4) symbolic representation and musical output. Besides, groundtruth construction and performance evaluation are also included.

## 3. MULTIMODAL AND MULTIMEDIA APPLICATION

In sheet music browser, the sheet music is located by different indexing methods and displays in a window with title on the top. Three kinds of indexing methods are provided: 1) by the “keywords” of song title; 2) by number of “strokes” of the first character of song title; 3) by “table of contents”. This is the usage convention for a paper songbook. For multimodal accompaniment for sight-reading, the interface is similar to that of music player. The audio play back is synchronized with sheet music reading, in which the current location is highlighted with the cyan ‘^’ sign. Besides, the playing position could be changed by double clicking on sheet music. In the game mode, the melody is playing as the multimodal accompaniment do but with randomly generated note pitch and length errors waiting to be detected by the players.

The screen shot of the sheet music browser in Figure 1 (a) is shown with indexing method labels at left hand side. After indexing, the list menu will show all the candidate songs with the index method indicated at the left. When some item is clicked, the sheet music will appear for preview with scroll bar in the panel with title “Sheet Music Preview”.

In the singing accompaniment as the screen shot of the Figure 1 (b), the application synthesizes the note audio of sheet music with the specific tempo value indicated in the

*Copyright: © 2019 Fu-Hai Frank Wu et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.*



Figure 1. (a) Sheet music browser (b) multimodal accompanying (c) game for pitch and length error detection

slider. When playing, the “^” sign will indicate the current note as the melody is going. User could double click on the screen to change the playing position or change the playing tempo by the slider. Beside of playing, the play button also has the function to replay the song from the beginning of the sheet music.

In the game as the Figure 1 (c), some of notes are generated erroneously and randomly. The game player

clicks on the screen of sheet music to indicate the location of the erroneous notes. The background scoring agent will evaluate the clicking results and indicate the results on the sheet with “v” and “x” sign for correct and wrong ones, respectively. The agent also makes scoring on the top and central area. The scoring includes score(F-measure), hit and miss. The hit item has two numbers separated by slash “/”. The number at left is hit number, and the number at right is the count of the generated errors.

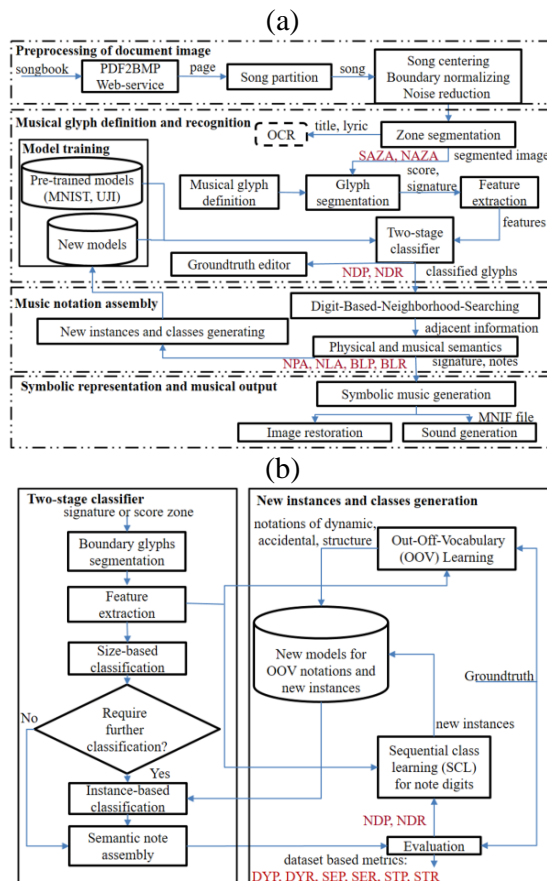


Figure 2. (a) OMR system flowchart with evaluation metrics in brown, (b) two-stage classification and learning for classes and instances

## 4. CONCLUSIONS AND FUTUREWORK

In the demo, we introduce our OMR ecosystem and demonstrate the application of multimodal and multimedia songbook. In order to have the automatic OMR tool, we empirically define the process flow and implement the prototype. Moreover, we took on the results of OMR to implement the multimodal interactive and multimedia application of songbook. To summarize, the research provides a promising approach to produce, distribute, and use sheet music joyfully.

In the future work, we could add the MIR functions, such as singing assessment, synchronization of audio source, score following based on some fundamental research achievement.

## Acknowledgments

The authors thank to the reviewer for the appreciation of the demo.

## 5. REFERENCES

- [1] Numbered musical notation, [http://en.wikipedia.org/wiki/Numbered\\_musical\\_notation](http://en.wikipedia.org/wiki/Numbered_musical_notation)
- [2] Fu-Hai Frank Wu, "Applying Machine Learning in Optical Music Recognition of Numbered Music Notation," International Journal of Multimedia Data Engineering and Management (IJMDEM) 8.3, pp 21-41, 2017.

# CAPTURING THE REACTION TIME TO DISTINGUISH BETWEEN VOICE AND MUSIC

**Alejandro Villena-Rodríguez, Lorenzo J. Tardón, Isabel Barbancho, Ana M. Barbancho, Irene Gómez-Plazas**

ATIC Research Group, Universidad de Málaga, Andalucía Tech, E.T.S.I. Telecomunicación,  
Dpto. Ingeniería de Comunicaciones, Campus Universitario de Teatinos s/n, 29071, Málaga, Spain  
avillennarod@uma.es, lorenzo@ic.uma.es, ibp@ic.uma.es, abp@ic.uma.es, irene.1996f1@uma.es

**María-José Varela-Salinas**

Universidad de Málaga, Andalucía Tech, Facultad de Filosofía y Letras,  
Dpto. Traducción e Interpretación, Campus Universitario de Teatinos s/n, 29071, Málaga, Spain  
mjvs@uma.es

## ABSTRACT

Reaction times (RTs) are an important source of information in experimental psychology and EEG data analysis. While simple auditory RT has been widely studied, response time when discriminating between two different auditory stimuli have not been determined yet. The purpose of this experiment is to measure the RT for the discrimination between two different auditory stimuli: speech and instrumental music.

## 1. INTRODUCTION

Reaction time (RT) is defined as the elapsed time between the presentation of a sensory stimulus and the subsequent behavioral response [1]. Simple auditory RT is one of the fastest RTs and is thought to be rarely less than 100 ms [2]. On the other hand, other studies show that the RT, when discriminating between two different stimuli, gets faster as the difference between the stimuli decreases [3].

The experiment considered in this demonstration consist of a go / no-go test where participants are asked to press a button when either a music piece or a speech excerpt is played.

As the stimuli are fairly different in this specific scenario, the RT for the recognition task is expected to be fast but, on the other hand, slower than known simple RTs for auditory stimuli. However, said RT is also expected to increase for speech excerpts presented in a language different from the participant's mother tongue.

## 2. STIMULI

The stimuli consist of a set of 50 instrumental music and 50 speech excerpts with duration of up to 4 seconds. The music excerpts are randomly cropped from the music tracks

Track name	Author
The Magic Flute	Wolfgang Amadeus Mozart
Emperor Waltz	Johann Strauss
Universal Mind	Liquid Tension Experiment
Adios Nonino	Astor Piazzola

Table 1. List of music pieces where the excerpts are taken from.

shown in Table 1. The presented pieces of speech consists of unconnected sentences recorded in a varied set of languages: Spanish, English, German, French and Danish. In order to keep the variance of volume at the minimum, all the excerpts are compressed and normalized. Each excerpt will be identified by means of pulse width modulation of a trigger/synchronization signal.

## 3. EQUIPMENT

Stimuli will be presented to the subjects through headphones (Sennheiser HD 219) using a system with Presentation software from Neurobehavioral Systems. The response action is recorded by pressing a single USB button connected to the system.

## 4. METHODOLOGY

In this section, the basis of the methodology employed in the experiments is presented.

### 4.1 Prior to the experiment

Attendants from the Sound and Music Computing Conference 2019 will be encouraged to participate as experimental subjects. The attendants who will volunteer will be asked to fill in a form with questions about parameters that can have an effect on the response time, such as their gender, age, mother tongue, known languages and conditional factors like fatigue and arousal [4].

Before the experiment begins, a royalty-free track called "Casual Friday" which is unrelated to the experiment will be played, with the objective of setting a comfortable hearing threshold for each participant.

Copyright: © 2019 Alejandro Villena-Rodríguez, Lorenzo J. Tardón, Isabel Barbancho, Ana M. Barbancho, Irene Gómez-Plazas et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



## 4.2 During the experiment

Once the subject is totally set up, four known test trials will be prompted in order to train the participant. Following these test trials, the music and speech excerpts will be presented to the subject in random order with a pause of 2 seconds between each excerpt.

The experiment, for each subject, will be carried out in 2 separate blocks of trials: in the first block, the subject will be asked to press the button only when a musical piece is played; in the second one, the button should be pressed by the subject only when a speech is played. Both blocks have the same amount of excerpts.

The estimated time for the experiment is 20 minutes plus the explaining and testing time.

## 5. CONCLUSIONS

The current community lacks of a proper set measurements of the RT when recognizing between speech and music. Also, since the speech excerpts are presented in different languages to a variety of nationalities, the results of this study are not constrained to a certain population with a common mother tongue but expected to be widely valid.

A large set of RTs, like the one described in this demonstration, will not only benefit future EEG studies but will also be helpful to the experimental psychology community or to any researcher interested in understanding how the music and speech is processed in the brain.

## Acknowledgments

This work has been funded by the Ministerio de Economía y Competitividad of the Spanish Government under Project No. TIN2016-75866-C3-2-R. This work has been done at Universidad de Málaga, Campus de Excelencia Internacional (CEI) Andalucía TECH.

## 6. REFERENCES

- [1] J. Shelton and G. P. Kumar, "Comparison between auditory and visual simple reaction times," *Neurosci Med*, vol. 1, no. 1, pp. 30–32, 2010.
- [2] M. T. Pain and A. Hibbs, "Sprint starts and the minimum auditory reaction time," *Journal of sports sciences*, vol. 25, no. 1, pp. 79–86, 2007.
- [3] M. H. Bornstein and N. O. Korda, "Discrimination and matching within and between hues measured by reaction times: Some implications for categorical perception and levels of information processing," *Psychological research*, vol. 46, no. 3, pp. 207–222, 1984.
- [4] R. H. Baayen and P. Milin, "Analyzing reaction times," *International Journal of Psychological Research*, vol. 3, no. 2, pp. 12–28, 2010.

# PHYSICAL MODELS AND REAL-TIME CONTROL WITH THE SENSEL MORPH

**Silvin Willemsen**  
Multisensory Experience Lab, CREATE,  
Aalborg University Copenhagen  
Copenhagen, Denmark  
sil@create.aau.dk

**Stefan Bilbao**  
Acoustics and Audio Group  
University of Edinburgh  
Edinburgh, UK  
s.bilbao@ed.ac.uk

**Nikolaj Andersson, Stefania Serafin**  
Multisensory Experience Lab, CREATE,  
Aalborg University Copenhagen  
Copenhagen, Denmark  
{nsa, sts}@create.aau.dk

## ABSTRACT

In this demonstration we present novel physical models controlled by the SENSEL Morph interface.

## 1. INTRODUCTION

The SENSEL Morph is a highly accurate touch controller that senses position and force of objects [1]. Figure 1 shows one player interacting with two SENSEL Morph devices to interact with the developed physical models. We use the SENSEL as an expressive interface for interacting with different physical models described in a companion paper accepted to SMC 2019. Right above the touch-sensitive area, the SENSEL contains an array of 24 LEDs that can be controlled from the application.

Strings are shown as coloured paths (see Figure 2 for a descriptive visualisation). The state of the string is visualised using the vertical displacement of the paths. Bowed strings are shown in cyan on the top left. The bow is shown as a yellow rectangle and moves while interacting. The fretting position is shown as a yellow circle. Plucked strings are shown in purple in the top right, underneath which the sympathetic strings are shown in light green. The plate is shown in the bottom using a grid of rectangles (clamped grid points are not shown). Its state is visualised using a grey-scale. Furthermore, connections are shown using orange circles/squares for the points of connection and dotted lines between these points. Lastly, all parameters that are controlled by the mouse such as output-level and plate-stiffness are located in a column on the right side of the application.

## 2. IMPLEMENTED INSTRUMENTS

We subdivide string-elements into three types: bowed, plucked and sympathetic strings. All strings will be connected to one plate acting as an instrument body of which the user can control the plate-stiffness. Furthermore, the user can change the output-level of each element type. Apart from these parameters, which are controlled by the mouse, the

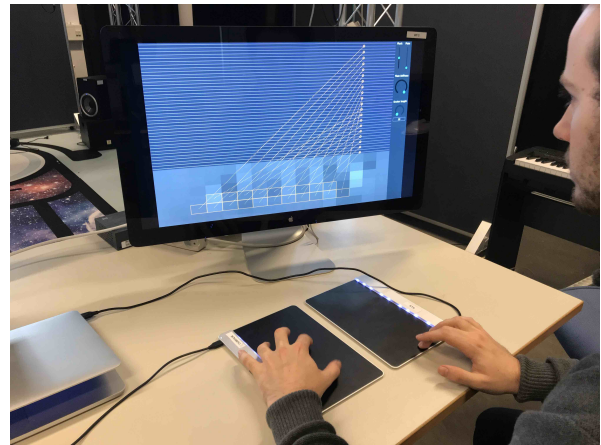


Figure 1. Player using the SENSEL Morphs to interact with one of the instruments.

instruments are fully controlled by two SENSELs. The instruments we have chosen as our inspiration are the sitar, the hammered dulcimer and the hurdy gurdy.

### 2.1 Bowed Sitar

The sitar is originally an Indian string instrument that has both fretted strings and sympathetic strings. Instead of plucking the fretted strings, we extended the model to bow them. Our implementation consists of 2 bowed strings (tuned to A3 and E4), 13 sympathetic strings (tuned according to [2]) and 5 plucked strings (tuned A3-E4 following an A-major scale) as it is also possible to strum the sympathetic strings. Figure 2 shows the visual interface of the implementation. One SENSEL is vertically subdivided into two sections; one for each bowed string. The first finger registered by the SENSEL is mapped to a bow and the second is mapped to a fretting finger controlling pitch. The horizontal position of both fingers is visualised using the SENSEL's LED array. The frets are not implemented as such (the pitch is continuous), but they are visualised for reference.

### 2.2 Hammered Dulcimer

The hammered dulcimer is an instrument that can be seen as an 'open piano' where the musician has the hammers in their hand. Just like the piano, the strings are grouped in

Copyright: © 2019 Silvin Willemsen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

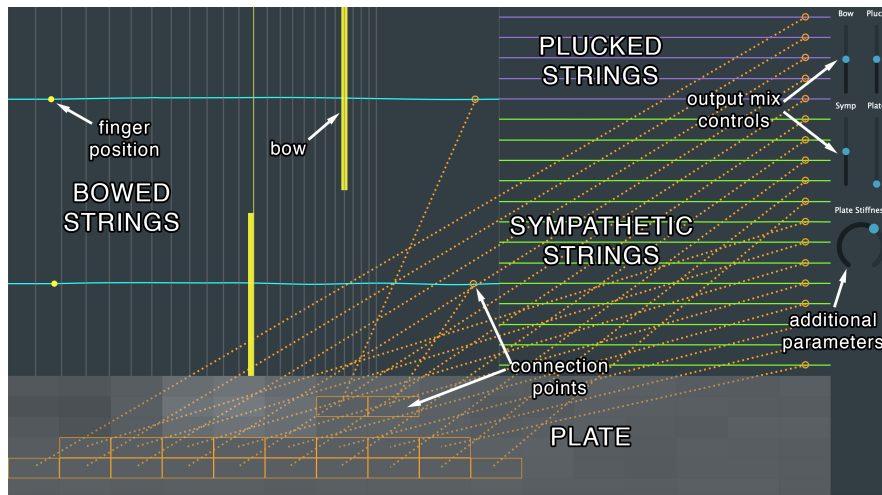


Figure 2. The bowed sitar application. The descriptions of the different elements and other objects are shown in the image, but will (naturally) not be visible in the application.

pairs or triplets that are played simultaneously. The interface for the hammered dulcimer can be seen in Figure 3. In our implementation, we have 20 pairs of plucked strings. Even though most hammered dulcimers have more strings, we decided that this configuration has the highest number of strings while maintaining playability. One of each pair is connected to a plate which slightly detunes it, creating a desired ‘chorusing’ effect. Two Sensel boards are placed vertically next to each other (see Figure 1). The pair with the lowest frequency is located in the bottom right and the highest in the top left, as in the real instrument. As with the plucked strings of the bowed sitar, the LED array is used to visualise the way that the Sensel is subdivided, which is especially useful here as one Sensel controls 10 string-pairs.

The mass ratio is set relatively high ( $\mathcal{M} = 100$ ) to amplify the non-linear interaction between the strings and the detuning of the strings connected to the plate.

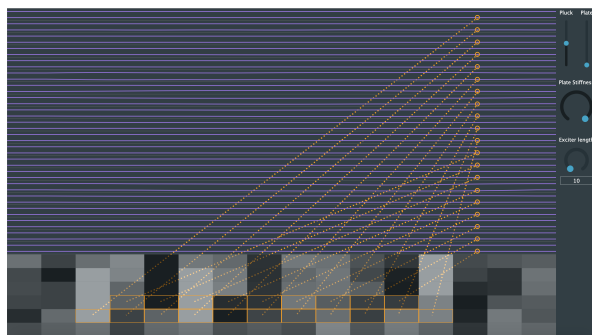


Figure 3. The hammered dulcimer application.

### 2.3 Hurdy Gurdy

The hurdy gurdy is an instrument that consists of bowed and sympathetic strings. The bowing happens through a rosined wheel attached to a crank and bows these strings as the crank is turned. It is possible to change the pitch

of a few bowed strings - the melody strings - using buttons that press tangent pins on the strings at different positions. The other strings, referred to as drone strings, are mostly tuned lower than the melody strings and provide the bass frequencies of the instrument. The musician can place the bowed strings on rests that keep the wheel from interacting with it. The visual interface can be seen in Figure 4. Our implementation consists of 5 bowed strings subdivided into 2 drone strings tuned to A2, E3 and 3 melody strings tuned to A3, E4 and A4 and 13 sympathetic strings tuned the same way as the sympathetic strings in bowed sitar. Furthermore, the mass ratios have been set the same as in the bowed sitar application.

The Sensel is vertically subdivided into 5 rows that control whether the strings are placed on the wheel.

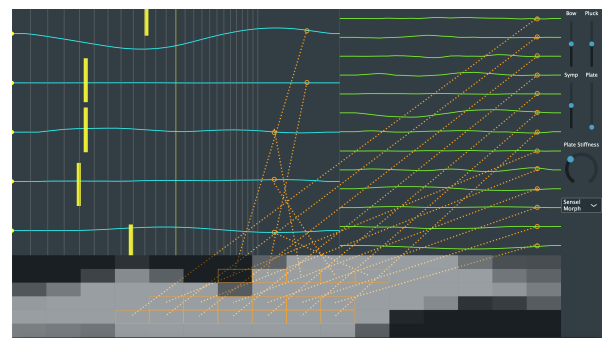


Figure 4. The hurdy gurdy application.

### 3. REFERENCES

- [1] Sensel Inc. (2018) Sensel morph. [Online]. Available: <https://sensel.com/>
- [2] Joe Rizzo. (2015) How to tune a sitar. [Online]. Available: <http://www.joerizzo.com/sitar/>

# Towards a High-Performance Platform for Sonic Interaction Interfaces

**Stefano Fasciani**

Faculty of Engineering and Information Sciences,  
University of Wollongong in Dubai  
Department of Musicology,  
University of Oslo  
stefano@fasciani.xyz

**Manohar Vohra**

Faculty of Engineering and Information Sciences,  
University of Wollongong in Dubai  
mv800@uowmail.edu.au

## ABSTRACT

In this paper we introduce a hardware platform to prototype interfaces of demanding sonic interactive systems. We target applications featuring a large array of analog sensors requiring data acquisition and transmission to computers at fast rates, with low latency, and high bandwidth. This work is part of an ongoing project which aims to provide designers with a cost effective and accessible platform for fast prototyping of complex interfaces for sonic interactive systems or musical instruments. The high performances are guaranteed by a SoC FPGA. The functionality of the platform can be customized without requiring significant technical expertise. In this paper, we discuss the principles, the current design, and the preliminary evaluation against common microcontroller-based platforms. The proposed platform can sample up to 96 analog channels at rates up to 24 kHz and stream the data via UDP to computers with a sub millisecond latency.

## 1. INTRODUCTION

The user interface is an essential component in sonic interactive systems. The manipulation of such an interface determines aspects of the sound generation process that in turn affects the user's manipulation [1]. The closed-loop architecture requires tight coupling, or low latency, between action and auditory feedback. This requirement holds across a large spectrum of applications, including interactive sonic installations, electronic musical instruments, Virtual Reality (VR) and videogames.

Independent of the input modality, these interfaces integrate sensors to transduce the user's gesture into electric signals. Features extracted from these signals control parameters of the sound synthesis by explicit or generative mapping techniques [2]. However, before extracting features, the analog signals must be sampled, digitized, and then transferred to a computational system. The platform we introduce in this paper is designed to carry out these tasks in context to requiring large number of channels, a high sampling rate, low latency and a large bandwidth towards the sound synthesis module.

When prototyping sonic interactive systems, the hardware for acquisition of analog signals is often taken off-the-shelf due to a lack of time or expertise. Moreover, choices are often oriented towards platforms that are easy to customize and integrate, even if it is not specifically designed for this application domain. This allows focusing on gesture-capturing, mapping, sound synthesis and

creative applications, which are core components of sonic interaction studies. However, these platforms deliver poor sampling rates and latency with a large number of sensors, not to mention the incapability of simultaneous sampling of signals. The detrimental consequences can significantly deteriorate the resulting sonic feedback and the overall user experience.

High-end data acquisition systems that can provide sufficient performances are available on the market, but these are costly and cannot be customized. Moreover, their latency is often undocumented or excessive as they are not designed for sonic interactive applications. The development of such data acquisition system requires a significant engineering effort which is not suitable in resource-constrained projects, or when rapid prototyping is needed. The platform we describe here enables fast-prototyping of interfaces for demanding sonic interaction systems. We selected a cost-effective, yet powerful platform based on a System on Chip (SoC) Field Programmable Gate Array (FPGA) which can be customized by designers without modifying the internal architecture.

## 2. EXISTING PLATFORMS

In the last decade we assisted to the proliferation of microcontroller-based boards and single-board computers that made prototyping of interactive systems accessible to everyone. The cost of these boards is significantly lower than those produced by silicon manufacturers. In addition, integrated development tools, rich sets of libraries, online user communities, and the open-source nature have drastically simplified the development workflow.

### 2.1 Microcontroller-based Platforms

The concept of accessible development boards was pioneered by Arduino. Introduced in 2005, this platform aimed at reducing cost of student's projects at the Interaction Design Institute Ivrea, and providing an easy-to-use Integrated Development Environment (IDE) [3]. The popularity and pedagogical value of Arduino was also determined by the simplified approach to introduce complex hardware and software concepts in their tutorial and example projects [4]. This enabled novices to implement embedded systems for physical computing in a relatively short amount of time. Most Arduino boards are based on 8-bit AVR microcontrollers. The popular Arduino Uno, released in 2010, is equipped with an ATmega328P running at 16 MHz. Thereafter, a variety of compatible expansion boards (or shields), and clone or compatible

Copyright: 2019 Stefano Fasciani et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



boards have been commercialized. These support the same Application Programming Interface (API), Hardware Abstraction Library (HAL), libraries and IDE of the original Arduino boards, making programs easier to port across platforms. Interaction designers can choose among a wide spectrum of easy-to-use boards with different size, computational power, interfacing capabilities, and price. An increasing number of Arduino compatible boards feature an ARM Cortex-M 32-bit microcontroller. These run at higher clock rates than the AVR, provide native 32-bit computation, a nested vector interrupt controller and a richer set of I/O.

Despite the relatively high-level API, programming in Arduino IDE is generally bare-metal. Libraries supporting threads and simple real-time Operating Systems (OS) that do not require Memory Management Units (MMU), such as FreeRTOS [5], are available for both AVR and ARM Cortex-M architectures. Easy-to-use libraries for sound synthesis, such as Mozzi [6] are available to designers. Firmata [7] further minimizes the programming burden for designers using Arduino compatible boards only to capture sensors data and transfer it to a computer. Firmata bundles a microcontroller program, a communication protocol, and clients for programming environments including a Graphical User Interface (GUI) to configure the physical I/O.

A more recent platform that is gaining popularity among interaction designers is ARM Mbed, a collaborative project managed by ARM Holdings. ARM Mbed boards are cost effective and produced by several manufacturers, including major semiconductor companies. These boards are based on the ARM Cortex-M architecture (a few boards mount a Cortex-A), sharing the same API, HAL and IDE. Compared to the Arduino, ARM Mbed provides a more sophisticated OS and software libraries, along with powerful microcontrollers (clock rates between 30 and 200 MHz) and a richer set of peripherals. In terms of programming complexity, ARM Mbed is as accessible and well supported as Arduino. C libraries for sound synthesis can be easily ported to Mbed platforms, such as OOPS [8].

Although Arduino and ARM Mbed are relatively easy-to-use platforms, but the development of a complete sonic interactive system on these platforms is challenging due to a lack of high-level programming languages and OS. Indeed, these are generally used to capture sensor data and transfer it to computers, where feature extraction, mapping and synthesis can be easily implemented using high-level programming environments. Critical information such as maximum sampling rate on multiple analog inputs, synchronicity, bandwidth, and latency of communication channels are not available nor can be estimated from the documentation. Hence it is difficult to select a platform matching the design requirements.

## 2.2 Single-board Computers

To date, a large variety of single-board computers are available in the market [9]. The majority are equipped with a single or multi-core 32-bit ARM Cortex-A with MMU and clock rates typically within 1 to 2 GHz. Recent boards are switching to 64-bit ARM Cortex-A archi-

tectures. Most boards support at least one Linux distribution, whereas Android, Windows CE and BSD are other common OSs. The most popular are the Raspberry Pi and the Beagleboard. Both are affordable, well supported and relatively simple to use. Both support Debian OS and provide an HDMI video output, hence the development workflow on these platforms is much alike to general-purpose computers. For instance, it is possible to use high-level audio programming environment such as Pure Data, FAUST and Csound.

The computational power of these single-board computers is suitable to implement simple end-to-end sonic interactive system, from gestural signal acquisition to sound synthesis, although only a minority of these provides on-board Analog to Digital Converter (ADC). Also, response latency and jitter of these platforms do not meet the minimum requirement of most sonic interactive systems. These issues were addressed by Satellite CCRMA [10], and particularly by Bela [11]. Bela integrates an expansion board with multiple analog I/O, a customized real-time operating system, and an audio driver delivering a sub-millisecond gesture-to-sound latency. It provides a stereo audio input and output with a sampling rate of 44.1 kHz and 16-bit resolution, 8 analog inputs and 8 analog outputs for sensors and actuators with a sampling rate of 22.05 kHz and 16-bit. Using a dedicated multiplexer board, the number of analog inputs can be expanded to 64 with 2.75 kHz sampling rate.

Single-board computers provide only a fraction of the computational power and memory available on general-purpose computers, hence the implementation of demanding sonic interactive applications may not be possible. Using these boards only to capture sensor data and transmit it to a computer is wasteful in term of computational resources, while latency and jitter are higher than microcontroller-based boards due to the OS.

## 2.3 FPGA-based Platforms

FPGAs can meet demanding requirements with respect to throughput and latency. Recent technological advancements have enabled the manufacturing of powerful but relatively low-cost and low-power FPGA chips [12]. The Xilinx 7<sup>th</sup> Series FPGAs are featured on Digilent boards and are available for approximately 100 USD, including Spartan, Artix and Zynq FPGAs [13], [14]. Tools and languages for FPGA development have significantly improved, but the workflow to design and implement an FPGA-based system still requires significant engineering expertise. Platforms such as Mojo, Papilo (both featuring an old Spartan 6), and Alchitry provided developers with a low-cost, small-size, and developer-friendly boards, including an Arduino-compatible connector. However, tools are still far from being accessible when compared to platforms discussed in the previous sections. Hardware Description Language (HDL) still represents as an entry barrier for most designers. Arduino has recently launched the MKR Vidor 4000, a board featuring a 32-bit ARM Cortex-M and a small Intel (formerly Altera) Cyclone 10 FPGA. The company has announced, but not yet disclosed, a promising and easy-to-use online visual programming tool to program the chip.

To date, only a handful of sonic interactive systems or New Interfaces for Musical Expression (NIME) feature an FPGA-based platform, such as Overholt’s Matrix [15], the continuous keyboard by Freed and Avizienis, the SLABS by Wessel, Freed and Avizienis [16], a physical modeling drum controller by Chuchaz, O’Modhrain and Woods [17], a controller for physical modeling synthesis by Pfeifle and Bader [18], and the Arcontinuo by Cadiz and Sylleros [19]. Freed, Avizienis, Wessel and Wright underlined necessity for high-performance interfaces (in terms of bandwidth and latency) between an array of sensors and computers almost two decades ago [20], [21]. A few FPGA-based platforms to capture gestural data from sensors have been proposed, such as SensorLab from Steim, the Connectivity Processor from CNMAT [22], and the Gluion [23]. Bandwidth and latency benchmark for the SensorLab and the Gluion are not available, but analyzing their technical features is evident how these cannot match the performances of the Connectivity Processor, that features 10 channels of 24-bit audio sampled at 48 kHz, 64 channels of sample-synchronous control-rate gesture data sampled at 6 kHz, and overall latency of 7 milliseconds with Max/MSP. None of these systems is on the market nor details for their fabrication are available.

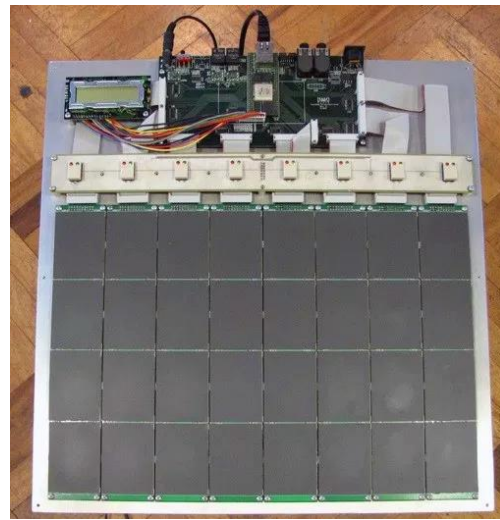
### 2.3.1 The SLABS

The SLABS is among the most complex and expressive musical interfaces ever built. The elegant and effective design is the result of almost a decade of development by Avizienis, Freed and Wessel at CNMAT. It features a matrix of touch-sensitive pads that can be mapped to computer software in a variety of ways. The SLABS “was designed to engage the body, to be both musically expressive and inspiring, to be easy to play at the entry level, and to be accepting of a lifelong development of virtuosity” [16]. Each pressure-sensitive touchpad produces three analog signals related to the touch coordinates and pressure. These are sampled at a high rate and sent to the computer via Ethernet as audio streams, enabling a tight coupling between gesture and data. Gestural data can be used directly as a signal within the synthesis algorithm or processed for the fast detection gestures. Wessel developed various Max/MSP patches interacting with the SLABS, including demos with simple oscillators, granular synthesis, and control of percussive loops.

Two versions of the SLABS have been fabricated, featuring respectively 24 and 32 VersaPad pressure-sensitive touchpads by Interlinks. The VersaPad signal acquisition hardware was modified to enable simultaneous sampling at high rate [24], [25]. In particular, the 96 analog signals (three from each touchpad) are first sampled at 6 kHz, then up-sampled to match the audio rate of 48 kHz and sent to the computer running Max/MSP as multichannel UDP audio stream. The SLABS can also transmit sensor data via Open Sound Control but at a much lower rate.

The core of the SLABS is the Xilinx Virtex-4 FX12 Evaluation Kit manufactured by Avnet. The module features an SX4VFX12-FF668 SoC FPGA with an embedded PowerPC core running at 300 MHz, 64 MB of DDR SRAM, 8 MB of flash memory, and a Gigabit Ethernet

interface. The module is installed on a custom motherboard, as visible at the top of Figure 1, that includes other I/O such as ADAT and MIDI, and the ADC chips sampling the signals from the touchpads. Samples are converted to 32-bit floating point before UDP transmission, therefore the bandwidth of the payload between the SLABS and the computer (excluding UDP framing) is approximately 147.5 Mbit/s. The cost of the FPGA module is approximately 450 USD. Details on the motherboard are not available, such as the model and resolution of the ADC chips, however, the layout of the board reveals 8 ADC chips sampling 12 signals each. We estimate the cost of the ADC chips to be about 150 USD.



**Figure 1.** The SLABS32, with motherboard and FPGA module visible on the top of the matrix of touchpads.

## 3. BENCHMARK OF MICROCONTROLLER-BASED PLATFORMS

As discussed in Section 2, microcontroller-based platforms are suitable and easy-to-use to acquire and transfer a small number of analog signals to a computer. From their documentation, it is not possible to determine the resulting latency and maximum sampling rate when acquiring a given number of analog signals. This is due to the complex interdependency between the software and hardware architectures. McPherson, Jack and Moro proposed a setup to measure latency and jitter on these platforms [26]. Their latency measurements include computer-based sound generation, showing how popular boards such as Arduino Uno and Teensy 2.0 struggle to achieve end-to-end latency below 10ms. These measurements were based on the acquisition of a single channel. The latency can increase with a larger number of inputs.

The benchmarks we present here focus on bandwidth and the measurement setup does not include a computer-based sound generation unit. In particular, we investigate the maximum rate at which multiple signals can be acquired and transferred to computers. We selected three boards: the Arduino Uno, the Teensy 3.6 by PJRC, and NUCLEO-F746ZG by ST Microelectronics. The Teensy 3.6 and the NUCLEO-F746ZG are respectively top-of-

the-line among Arduino and Cortex-M ARM Mbed boards. According to the specifications, the on-chip USB of the Teensy 3.6 microcontroller outperforms the one in the NUCLEO-F746ZG, while the on-chip Ethernet of the NUCLEO-F746ZG microcontroller outperform any Ethernet expansion module for the Teensy 3.6. For this reason, we did not carry out Ethernet-based benchmarks on the Teensy 3.6 and USB-based benchmarks on the NUCLEO-F746ZG. Links based on IEEE 802.11 wireless LAN standards were not considered due to their excessive latency. The benchmark programs use only standard libraries and API for Arduino and Mbed IDE available to average developers. Where possible, the programs configure the link and ADC to run at the maximum speed using simple instructions only. We did not use Direct Memory Access (DMA) to improve performances as this option is not available through standard API, but it requires microcontroller-specific expertise.

The Arduino Uno features an 8-bit AVR ATmega328P running at 16 MHz. We measured data transmission to the computer via USB through the onboard ATmega8U2 acting as a USB-to-Serial with a baud rate of 2 Mb/s, and via the 10/100 Ethernet through the W5500 Ethernet controller on the Ethernet Shield 2. We increased ADC conversion clock from the default 125 kHz to 8 MHz. The Uno can acquire up to 6 analog inputs multiplexed to a single bit 10-bit ADC, and up to 11 digital inputs (serial transceiver and LED lines are excluded).

The Teensy 3.6 features a 32-bit Freescale ARM Cortex-M4 processor with the Floating-Point Unit (FPU) running at 180 MHz. We measured data transmission to the computer through the on-chip USB peripheral configured as a USB-to-Serial with a baud rate of 4.608 Mb/s. We overclocked the Cortex-M4 to 240 MHz, and compiled the code using the Fastest optimization option. The Teensy 3.6 can acquire up to 24 analog inputs multiplexed to two 12-bit ADCs, and up to 55 digital inputs. Simultaneous sampling of two channels at a time is possible, but the available libraries provided worst performances than sequential sampling used in the benchmark.

The NUCLEO-F746ZG features a 32-bit ST Microelectronics ARM Cortex-M7 with FPU running at 216 MHz. We measured data transmission to a computer via User Datagram Protocol (UDP) through the on-chip 10/100 Ethernet Media Access Controller (MAC) with dedicated DMA. The NUCLEO-F746ZG can acquire up to 24 analog inputs multiplexed to three 12-bit ADCs, and up to 90 digital inputs. Simultaneous sampling of three channels at a time is possible only by complex low-level register programming and DMA. The simple Mbed API for sequential sampling was used within the benchmarks.

The benchmark programs repeatedly acquire a set of analog and/or digital inputs and sent a packet of data to the computer via USB or Ethernet, without any synchronization mechanism. The computer measures the average inter-arrival time between packets, reflecting the highest rate at which the board can sample and transmit the analog signals. The result of the ADC conversion (10 or 12-bits) is stored in a 16-bit integer, requiring the transmission of 2 bytes. The inter arrival time of packets was measured in Pure Data for USB link (Virtual COM Port), and in Wireshark for the Ethernet link. Details of the

benchmarks and results are summarized in Table 1, which shows the maximum rates when acquiring different set of analog and/or digital inputs. We evaluated the performances ranging from a single input to as many inputs as supported by the specific platform. Certain set were selected to facilitate comparison across platforms.

Platform	Link	Digital Inputs	Analog Inputs	Bytes Payload	Bytes Sent	Max Rate kHz
Arduino Uno	USB	1	0	1	1	53.7
		11	0	2	2	15.1
		11	0	11	11	7.1
		0	1	2	2	35
		0	6	12	12	6.2
		11	6	14	14	5.5
	Ethernet	1	0	1	60	2.3
		11	0	2	60	2
		11	0	11	60	2
		0	1	2	60	1.9
		0	6	12	60	0.9
		11	6	14	60	0.8
Teensy 3.6	USB	1	0	1	1	740
		11	0	2	2	378
		11	0	11	11	96.8
		36	0	5	5	222
		55	0	7	7	154
		0	1	2	2	127
		0	6	12	12	22.9
		0	24	48	48	5.8
		11	6	14	14	22
		36	24	53	53	5.6
		55	24	55	55	5.4
NUCLEO-F746ZG	Ethernet	1	0	1	60	17.5
		11	0	2	60	17.5
		11	0	11	60	17.5
		36	0	5	60	16.3
		55	0	7	60	15.8
		90	0	12	60	14.9
		0	1	2	60	15
		0	6	12	60	9.2
		0	24	48	90	3.8
		11	6	14	60	9.1
		36	24	53	95	3.8
		55	24	55	97	3.7
		90	24	59	101	3.6

**Table 1.** Benchmark details and results on maximum data acquisition rate for microcontroller-based platforms.

As expected, the maximum rates drop with the increase of the number of inputs. CPUs perform instructions sequentially, and the number of operations to perform increases linearly with the number of inputs. Further analysis demonstrated that the data transfer represents the bottleneck of these system, which keeps the CPU busy for a dominant fraction of the execution time. Getting the data from the ADC requires only a small fraction of the execution time. The aggregate sampling rates are significantly below the nominal maximum sampling rate of the ADCs of the platforms. Packing the data, especially for the digital inputs, present an overall advantage, as demonstrated in the tests with 11 digital inputs. When sending data via



Ethernet, we obtain poor performances with a small number of inputs due to the minimum payload of 18 data bytes in UDP packets, which requires additional 42 framing bytes. Buffering more samples per channel provide slightly higher rates but increase the latency.

For Arduino-compatible platforms, we also measured the maximum reading rates of Firmata [7] using the same baud rates and clock settings described above. Compared to previous benchmarks, Firmata presents a larger overhead when transmitting data over serial, since data is transmitted with an additional integer channel identifier, and samples from analog inputs are sent as 32-bit float numbers. By default, inputs are sampled approximately every 20 ms (50 Hz sampling rate), and this interval can be changed within the source code. Analog and digital data is transmitted to the computer only if the values are different from previously sent data. Therefore, the duration of one iteration of the loop can vary significantly, and signal sampling at regular interval cannot be guaranteed. On the Arduino Uno, Firmata supports up to 11 digital inputs and 6 analog inputs. Whereas on the Teensy 3.6, we have up to 36 digital inputs and 16 analog inputs. To benchmark the maximum rate of change that Firmata can handle on digital inputs, we drove these pins with a square wave and disabled analog input data acquisition. We measured the respective rate of change in the Firmata client for Pure Data. We increased the frequency of the square wave until measurement mismatching was above 1%. We repeated this measurement driving respectively 1, 11, and one 36 digital inputs (Teensy 3.6 only). In a separate benchmark we measured the maximum rate at which Firmata can acquire analog inputs, which were connected to a white noise source, while digital input acquisition was disabled. The inter-arrival time of packet of samples was still measured in the Firmata client for Pure Data. In both benchmarks, the sampling interval was set to 0 ms, forcing the data acquisition loop to run as fast as possible. Results are summarized in Table 2, where rates are rounded down to the nearest integer. For the digital inputs, the maximum rate is twice as the highest frequency of the square wave that the platform can acquire, taking one reading on both low and high levels. For the analog inputs, we can sample signals with spectral components up to half of maximum rates.

Platform	Inputs	Max Rate
Arduino Uno	1 digital	24 kHz
	11 digital	14 kHz
	6 analog	4 kHz
Teensy 3.6	1 digital	108 kHz
	11 digital	78 kHz
	36 digital	28 kHz
	6 analog	18 kHz
	16 analog	7 kHz

**Table 2.** Benchmarks on maximum data acquisition rate for analog and digital inputs using Firmata.

All considered platforms feature Successive Approximation Register (SAR) ADCs, which contributes to minimize the latency from acquiring the electrical signal to completing the associated data transmission. The latency from signal acquisition to data transmission was not ex-

plicitly measured. However, since the CPU iterates on a branch-free sequential loop that includes acquisition, data packing and transmission, the latency is at most one sampling period. This is significantly less than 1 ms and can be further reduced by interleaving data acquisition and data transfer when acquiring multiple channels. However, this can slightly worsen the maximum acquisition rate.

#### 4. HIGH-PERFORMANCE PLATFORM PRINCIPLES AND DESIGN

The systems we reviewed in Section 2 can sample multiple gestural signals with high temporal resolution. Gestural samples are locked to the audio sampling clock. This synchronous approach provides jitter free encoding of the gestures resulting in more control intimacy. Indeed, Wessel argues that “the high-bandwidth approach is the future when it comes to ultra-expressive electronic instruments because it allows so much performance data to be captured”. McPherson and Zappi state that using high sampling rates enables “capturing subtle details like audio-rate vibrations or detailed temporal profiles within sensor signals” [27]. The development of customized data acquisition systems with such characteristics is time consuming and requires significant expertise. Hardware cost, especially with flagship FPGAs from the Virtex family, are likely to exceed 1000 USD. Maintaining and preserving these complex artifacts is also a challenge, due to the rapid obsolescence of hardware and software [28].

The aim of this work is to provide designers with a ready-to-use platform, which is cost effective and simple to customize to meet specific application requirements. The platform supports a common design pattern where the hardware gesture controller is attached by a high-speed link to a computer running a media programming language such as Max/MSP or Pure Data. The principles we followed in our design include:

- FPGA-based platform with core functionalities implemented in hardware minimizing latency and jitter.
- Number of analog inputs and sampling rate at least matching the data acquisition system of the SLABS.
- Performances independent of the number of inputs.
- Set of acquisition expansion board supporting simultaneous or sequential sampling of the analog inputs.
- Filter bank for fast signal processing on board.
- Flexible and easy-to-configure number of inputs, sampling rate, buffer size, and filter coefficients.
- Cost in the range of 150 to 250 USD.

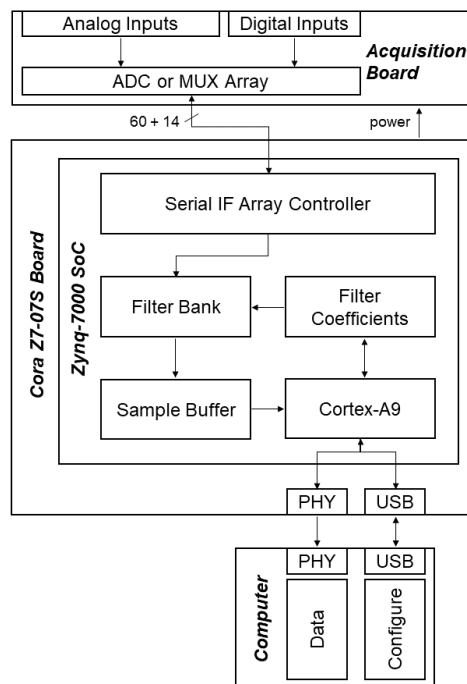
The simultaneous sampling of a large group of analog signals is costly because it requires one ADC per input (or at least one sample and hold circuit per input). Simultaneous sampling is necessary when phase information exists between different signals. Otherwise, it’s possible to use a fast ADC and analog multiplexers to sample sequentially all inputs but losing the phase information between signals. As platform designers, we cannot predict if preserving the phase relationship between signals would be required in the final application. However, this has a significant impact on the acquisition hardware and cost. Therefore, as detailed in Section 4.2, we propose three



acquisition subsystems: fully simultaneous, fully sequential, and a tradeoff between the two. For instance, in presence of multiple pressure-sensitive touchpads, sampling simultaneously three signals at a time is a tradeoff between cost and performance.

#### 4.1 System Architecture

The platform we selected to implement the system is the Digilent Cora Z7-07S [29], which features an Zynq-7000 SoC including an FPGA and an ARM Cortex-A9 processor running at 667 MHz, on board 1 Gbps Ethernet PHY and USB-UART bridge, priced at 99 USD. The proposed system architecture is illustrated in Figure 2.



**Figure 2.** Architecture of the proposed platform, with FPGA-based board and dedicated data acquisition board.

The platform communicates with one or more computers using Ethernet and USB (Virtual COM Port). These interfaces are handled by a bare metal program running the Cortex-A9, providing a predictable timed execution. The Ethernet only transmits packets of acquired data to a client. Control and configuration messages are exchanged with the computer asynchronously via USB. Configuration is allowed only when the data acquisition is not running. This approach maximizes the packet rate and minimize the jitter. Control and configuration messages include enabling/disabling data acquisition and transmission, destination IP address and port, selection of acquired inputs to be transmitted, sampling rate, buffer size and filter coefficients. The ADC and/or multiplexer chips on the acquisition board are handled by an array of dedicated serial interfaces. The data pass through a bank of biquad filters, and finally gets packed into buffers of samples. An optional stage can convert the sampled 16-bit integer to 32-bit floating point within the unitary range, aligned with representation of most computer-

based applications. The conversion to float will double the required Ethernet bandwidth, but it reduces the workload on the computer side. Serial interfaces, filters, and buffering are implemented in the FPGA fabric. On the Cora Z7-07S board, there are up to 74 FPGA pins exposed through stackable connectors, and 14 of these can be internally routed to the on-chip ADC. These are used in different configurations to interface in parallel the chips on the acquisition boards. This aspect is the bottleneck when using microcontrollers, as the number of serial interfaces cannot be expanded, and their control is usually sequential. Instead, the FPGA fabric allows implementing a large number of serial interfaces running in parallel.

#### 4.2 Acquisition Boards

When simultaneous sampling of a large set of analog signals is not necessary, we can use the XADC of the Zynq-7000 FPGA, which is a dual channel 1 Mega Sample Per Second (MSPS) SAR ADC with 12-bit resolution. Generally, this is accurate enough to sample non-audio analog signals from sensors. The Cora z7-07S connectors allow connecting 6 single ended and 4 differential inputs to the on-chip multiplexer of the XADC. These 14 lines are connected to the acquisition board and configured to 10 single ended analog inputs with reference voltage of 3.3 V. Each single ended input is connected to a high-speed analog multiplexer such as the MAX4617 from Maxim Integrated, obtaining a total of 80 analog inputs combining internal and external multiplexers. The 8 external multiplexers are controlled by 4 lines from the FPGA. The XADC and the multiplexer are controlled in hardware. When a single channel of the XADC is used, we achieved almost 200 kHz sampling rate on 10 inputs. The performance scales down almost linearly when using the external multiplexers, obtaining up to 24 kHz on 80 analog inputs. If using both converted of the XADC, the resulting rate does not improve significantly, but pairs of signals can be acquired in parallel. The remaining 56 FPGA pins can be used as digital inputs. The total cost of the components for this board is approximately 20 USD.

The second acquisition board we propose is based on the AD7991 from Analog Device, which is a non-simultaneous 4-channel single-ended 12-bit SAR ADC with an I2C interface. The AD7991 is also featured on the Digilent Pmod AD2, that we used for our measurements. When using the maximum serial clock of 3.4 MHz, it is possible to acquire one sample per channel at a rate slightly above 24 kHz. With 24 AD7991 we can obtain a total of 96 analog input, where groups of 24 inputs are sampled simultaneously. This requires 24 I2C controllers implemented on the FPGA fabric and a total of 48 FPGA pins. This configuration allows using the remaining 26 FPGA pins as digital inputs. The total cost of the components for this board is approximately 150 USD. Alternatively, we can use 12 AD7699 sampling simultaneously only 12 inputs but increasing the resolution to 16-bit. The number of used FPGA pins used is unchanged while the cost of the components increases to 180 USD.

Finally, to sample all 96 analog inputs simultaneously, the acquisition board can be based on the LTC2320-16

from Linear Technology, which is a 1.5 MSPS single-ended 8 channels simultaneous sampling SAR ADC with 16-bit resolution. To acquire 96 analog inputs, we require 12 LTC2320-16 interfaced using 32 FPGA pins, and 8 fast SPI interfaces implemented in the FPGA fabric. These chips enable sampling all channels simultaneously at 24 kHz, but also at the audio rate of 48 kHz. This configuration allows using the remaining 42 FPGA pins as digital inputs. The total cost of the components to implement this high-end board is approximately 240 USD.

### 4.3 Current Prototype and Performances

With multiple serial interfaces communicating with the ADC chips, the bottleneck of the architecture in Figure 2 is the Ethernet link communicating with the computer. In preliminary tests, we measured the bandwidth of the Ethernet link driven by a bare metal program based on the Xilinx porting of the LightWeight IP library and running on the Cortex-A9. The benchmarks are representative of different scenarios varying the number of analog inputs and the size of the sample buffer. Settings and results are summarized in Table 3.

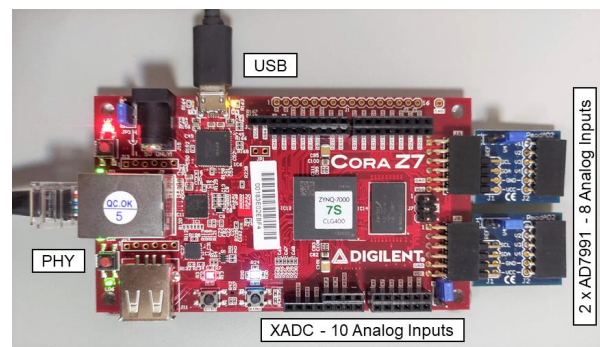
Analog Inputs	Buffer Size	Bytes Payload	Packet Rate kHz	Sampling rate kHz
18	1	36	84.5	84.5
	2	72	81.5	163.1
	4	144	72.4	289.8
	8	288	70.3	562.4
	16	576	61.1	977.5
	32	1152	49.4	1582.6
	64	2304	31.5	2017.5
34	1	68	81.1	81.1
	2	136	85.1	170.1
	4	272	63.6	254.4
	8	544	72.9	583.3
	16	1088	50.2	803.9
	32	2176	35.5	1138.1
	64	4352	15.3	985.5
96	1	192	68.7	68.7
	2	384	74.1	148.3
	4	768	60.1	240.5
	8	1536	34.6	277.1
	16	3072	27.1	434.9
	32	6144	12.9	413.0
	64	12288	6.4	410.4

**Table 3.** Benchmarks on Ethernet link bandwidth.

In the benchmarks of Table 3, independent of the ADC resolution, each sample is transmitted using 2 bytes. The payload of UDP packets does not include 4 bytes used to communicate with the client on the computer, and up to 5 additional bytes to transmit the data from digital inputs. When the payload is above 512 bytes, the data is split over multiple UDP packets. The results in Table 3 shows that the Ethernet PHY of Cora Z7-07S driven by the Cortex-A9 can easily accommodate the target sampling rates

of 24 kHz or 48 kHz. Moreover, it is also possible to accommodate double payloads we obtain when performing the integer to float conversion in the FPGA, transmitting 4 bytes per sample. With no buffering, we observed that the inter arrival packet time of UDP packets on the computer was affected by a jitter of up to 4 sampling periods due to the computer OS controlling the Ethernet.

All components in the architecture of Figure 2 have been individually tested and were verified that they meet the required performances. Moreover, we estimate that the FPGA on the Cora Z7-07S board has sufficient resources to implement the filter bank and array of serial controllers. Currently, we developed two complete prototypes of the system. The first is representative of the acquisition board with multiplexers. The board has been fabricated but most functionalities are still implemented on the Cortex-A9, which can deliver a maximum sampling rate of 16 kHz on 80 analog inputs. In the second prototype, we have two AD7991 controlled via two I2C with a serial clock of only 1 MHz. Additionally, we also take the samples from the XADC for a total of 18 analog inputs. The prototype uses four Pmod AD2 modules, as visible in Figure 3. Most functionalities are integrated in the Cortex-A9 and we obtained a maximum sampling rate of approximately 4 kHz. Settings such as the buffer size and enabling/disabling the data acquisition are available through on-board buttons or the serial monitor. In both prototypes, the filter bank has not yet been integrated. On the computer side, we used Wireshark to monitor timeliness and correctness of the received UDP packets. The worst-case latency from the acquisition of the analog signal to the transmission of the UDP packet is equal to the sampling period multiplied by the buffer size.



**Figure 3.** Platform prototype based on Digilent Cora Z7-07S and two Pmod AD2 modules.

## 5. FUTURE WORK

In this paper, we presented a platform to implement a complex interface for sonic interactive systems or musical instruments. The platform is capable of acquiring and transmitting a large number of analog signals from sensors to a computer, at rates significantly higher than microcontroller-based platforms. We designed the platform keeping in mind prototyping time and cost. Users can customize the functionality through simple commands without the need of developing any hardware or software components. The preliminary result of this ongoing pro-

ject shows that it is possible to achieve the target performance with the proposed architecture and acquisition boards, outperforming all microcontroller-based boards. In the immediate future, we will complete the schematics and the fabrication of the second and third acquisition board. Thereafter we will integrate the filter bank and complete the software for the computer, such as clients to receive the data and configure the platform will be developed as well. Sending a subset of the acquired analog signals to separate IP addresses via UDP is a possibility we will further explore. Moreover, we will consider exploiting unused FPGA resources to drive a DAC or generate Pulse Width Modulated (PWM) signals to drive actuators, hence enabling duplex communication between the platform and the computer.

### Acknowledgments

This work is supported by grant from the University of Wollongong in Dubai (No. 2018-007-SFMB). The authors would like to thank Adrian Freed for providing additional information on the SLABS.

## 6. REFERENCES

- [1] K. Franinović and S. Serafin, *Sonic Interaction Design*. MIT Press, 2013.
- [2] A. Hunt and M. M. Wanderley, "Mapping performer parameters to synthesis engines," *Organised Sound*, vol. 7, no. 2, pp. 97–108, 2003.
- [3] D. Kushner, "The making of arduino," *IEEE Spectr.*, vol. 26, 2011.
- [4] M. Banz, *Getting Started with Arduino*, Ill. Sebastopol, CA: Make Books - Imprint of: O'Reilly Media, 2008.
- [5] R. Barry, *FreeRTOS reference manual: API functions and configuration options*. Real Time Engineers Limited, 2009.
- [6] T. Barrass, "Mozzi: Interactive Sound Synthesis on the Open Source Arduino Microprocessor," in *Proc. of ICMC 2013*, Perth, Australia, 2013.
- [7] H.-C. Steiner, "Firmata: Towards Making Microcontrollers Act Like Extensions of the Computer," in *Proc. of NIME 2009*, Pittsburgh, US, 2009, pp. 125–130.
- [8] M. Mulshine and J. Snyder, "OOPS- An Audio Synthesis Library in C for Embedded (and Other) Applications," in *Proc. of NIME 2017*, Copenhagen, Denmark, 2017.
- [9] "Comparison of single-board computers," *Wikipedia*. 27-Jan-2019.
- [10] E. Berdahl and W. Ju, "Satellite CCRMA: A musical interaction and sound synthesis platform," in *Proc. of NIME 2011*, Oslo, Norway, 2011.
- [11] G. Moro, A. Bin, R. H. Jack, C. Heinrichs, A. P. McPherson, and others, "Making high-performance embedded instruments with Bela and Pure Data," in *Proc. of Int. Conf. of Live Interfaces*, Brighton, UK, 2016.
- [12] S. M. Trimberger, "Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology," *Proc. IEEE*, vol. 103, no. 3, pp. 318–331, 2015.
- [13] Xilinx Inc., "7 Series FPGAs Overview." 2018.
- [14] Xilinx Inc., "7 Series Product Selection Guide." 2018.
- [15] D. Overholt, "The MATRIX: A Novel Controller for Musical Expression," in *Proc. of NIME 2001*, Singapore, Singapore, 2001, pp. 1–4.
- [16] D. Wessel, "SLABS: Arrays of Pressure Sensitive Touch Pads | CNMAT," 2009. .
- [17] K. Chuchacz, S. O'Modhrain, and R. Woods, "Physical Models and Musical Controllers: Designing a Novel Electronic Percussion Instrument," in *Proc. of NIME 2007*, New York, US, 2007, pp. 37–40.
- [18] F. Pfeifle and R. Bader, "Performance Controller for Physical Modelling FPGA Sound Synthesis of Musical Instruments," in *Sound - Perception - Performance*, Springer, Heidelberg, 2013, pp. 331–350.
- [19] R. F. Cádiz and A. Sylleros, "Arcontinuo: the instrument of change.," in *Proc. of NIME 2017*, Copenhagen, Denmark, 2017.
- [20] D. Wessel and M. Wright, "Problems and prospects for intimate musical control of computers," in *Proc. of ACM Computer-Human Interaction Workshop on New Interfaces for Musical Expression*, Seattle, USA, 2001.
- [21] A. Freed, R. Avizienis, and M. Wright, "Beyond 0-5V: expanding sensor integration architectures," in *Proc. of NIME 2006*, Paris, France, 2006, pp. 97–100.
- [22] R. Avizienis, A. Freed, T. Suzuki, and D. Wessel, "Scalable Connectivity Processor for Computer Music Performance Systems.," in *ICMC*, 2000.
- [23] S. Kartadinata, "The Gluion Advantages of an FPGA-based Sensor Interface," in *Proc. of the 2006 Conf. on NIME*, Paris, France, 2006, pp. 93–96.
- [24] D. Wessel, R. Avizienis, A. Freed, and M. Wright, "A Force Sensitive Multi-Touch Array Supporting Multiple 2-D Musical Control Structures," in *Proc. of NIME 2007*, New York, US, 2007.
- [25] A. Freed, "Novel and Forgotten Current-steering Techniques for Resistive Multitouch, Duotouch, and Polytouch Position Sensing with Pressure," in *Proc. of NIME 2009*, Pittsburgh, US, 2009.
- [26] A. P. McPherson, R. H. Jack, and G. Moro, "Action-Sound Latency: Are Our Tools Fast Enough?," in *Proc. of NIME 2016*, Brisbane Australia, 2016.
- [27] A. McPherson and V. Zappi, "An environment for submillisecond-latency audio and sensor processing on BeagleBone Black," in *Audio Engineering Society Convention 138*, Warsaw, Poland, 2015.
- [28] A. Freed, "David Wessel's Slabs: a case study in Preventative Digital Musical Instrument Conservation," in *Proc. of SMC2016*, Hamburg, Germany, 2016.
- [29] Digilent Inc., "Cora 7Z Reference Manual." 2018.

# DIGITAL MANUFACTURING FOR MUSICAL APPLICATIONS: A SURVEY OF CURRENT STATUS AND FUTURE OUTLOOK

Doga Cavdir

CCRMA, Stanford University

CA 94305-8180, USA

cavdir@ccrma.stanford.edu

## ABSTRACT

In the design of new musical instruments, from acoustic to digital, merging conventional methods with new technologies has been one of the most commonly adopted approaches. Incorporation of prior design expertise with experimental or sometimes industrial methods suggests new directions in both design for musical expression and development of new manufacturing tools.

This paper describes key concepts of digital manufacturing processes in musical instrument design. It provides a review of current manufacturing techniques which are commonly used to create new musical interfaces and discusses future directions of digital fabrication which are applicable to numerous areas in music research, such as digital musical instrument (DMI) design, interaction design, acoustics, performance studies, and education. Additionally, the increasing availability of digital manufacturing tools and fabrication labs all around the world make these processes an integral part of the design and music classes. Examples of digital fabrication labs and manufacturing techniques used in education for student groups whose age ranges from elementary to university level are presented. In the context of this paper, it is important to consider how the growing fabrication technology will influence the design and fabrication of musical instruments, as well as what forms of new interaction methods and aesthetics might emerge.

## 1. INTRODUCTION

The musical instrument design process requires numerous artistic, musical and engineering design specifications from software design to electronics, from mechanical functionality of instruments and fabrication to compositions and performances. Cather et. al. states that “The lack of a complete and thorough written specification is now generally accepted as being one of the main reasons for design failure” [1, 2]. Commonly, the physical design, mechanical functionality, and rapid manufacturing concerns do not always become the priority of the instrument designers while new musical instruments experience limitations in fulfilling their purposes in the long term. These in-

struments, even though they are designed for professional use, most of the time, become restricted to be performed in research labs, demos, and recording sessions [3]. In order to increase their professional and artistic use in the long term, designers aim to embody many qualities of traditional instruments in new interfaces with existing fabrication and rapid prototyping techniques. As a consequence, the growth and advancement in digital fabrication techniques create a dynamic interaction between the research, arts, design and education fields.

Efforts in musical instrument design with digital manufacturing are divided into two areas; one performed by those experimenting new fabrication tools to manufacture acoustic instruments, and the other mostly explored by researchers who develop new interfaces, digital, augmented, or hybrid musical instruments. In the first case, the craftsmanship and knowledge on musical acoustics are crucial; yet, the instrument making process still requires other skills that relate to musical expression, aesthetics, and the interaction. On the other hand, researchers either leverage existing manufacturing techniques in unique ways for musical purposes or they develop new tools and advance the technology for higher quality instruments and interactions. This leads to an increasing demand for fast and accurate prototyping tools in designing new interfaces for musical expression.

This paper offers an overview of the current digital manufacturing techniques used in musical instrument design and discusses the application areas of the emerging fabrication tools in new musical expression. The next section, *Section 2*, introduces the field of digital fabrication giving prior examples from researchers, instrument makers, and musicians. *Section 3* discusses the future direction of digital fabrication as well as the possible application areas of the existing technology which is still unknown or uncommon in music and instrument design research. It further summarizes the emerging use of digital manufacturing in music education, examples of fabrication labs and musical instrument design classes.

## 2. MANUFACTURING MUSICAL INSTRUMENT

### 2.1 Rapid Prototyping

#### 2.1.1 Additive Manufacturing

Additive manufacturing (AM), also known as 3D printing, is a form of rapid prototyping which creates 3D objects by applying materials layer upon layer [4]. AM, still a

Copyright: © 2019 Doga Cavdir. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



rapidly growing technology, is now available for personal and commercial use, as well as for research and education. The decrease in the cost of the technology also leads music researchers, interface designers, and instrument builders to adopt 3D printing into their design process, allowing customized utilization of these technologies.



Figure 1. Hovalin, the 3D printed violin [5].

Earlier examples of instrument manufacturing based on 3D printing are mainly limited to acoustic violin, guitar, and flute fabrications [5, 6]. Hovalin [5] is one of the earliest examples of the 3D printed instruments, which provided a sustainable violin model (Fig. 1). After a number of iterations, designers could propose a model that can be printed without any support structure. Since the layers are supported by the layers beneath them, a model with overhangs (structure with no support below) requires an additional 3D printing support structures to ensure a successful print. Printing an acoustic instrument without support as Hovalin designers suggested increases the production speed significantly. Due to the dimension limitations of the available tools, the instrument needed to be printed in seven different parts, three of which formed the instrument body. Contrary to Hovalin's assembly method of gluing soundboard parts to each other, the Modular Fiddle was printed in one piece by Openfab PDX [7] whose source files are still available for personal manufacturing purposes. According to the designer, because of the single-piece instrument body, the sound is louder and richer than the earlier versions; yet, the sound quality is still not compatible to the mediocre versions of wooden instruments.

Specifically, the earlier versions of 3D printed objects were likely to have certain drawbacks, such as limitations in acoustic qualities and durability. Due to the type of material used in printing, these instrument shapes were deformed over time. Instrument bodies that needed to be printed in multiple pieces due to the printer dimensions also resulted in significant timbral changes. Previously, these technologies could only offer a limited selection of materials which directly affects the acoustics of printed instruments, as well as its tension resistance. An acoustic violin printed in the Formlabs offered black, white, and tough resin, new materials which brought a more stable structure and a cleaner finish. Yet these instruments printed in Formlabs still posed deformation problems of the neck warped under string tension [8]. Despite the drawbacks of the recent technology, these attempts made 3D printing technology available to the do-it-yourself (DIY) and maker communities, resulting in an increased in research of new materials, different printing patterns, and advanced

manufacturing technologies. Additionally, these attempts developed a maker community to share open source designs which can enhance personal manufacturing in the long term [9].

Relatedly, designing for AM comes in many different forms in terms of the variety of material, core technology, cost, or print time. Although the process extending from the design to the end product follows a similar path, 3D printing techniques differ in the technology behind them such as stereolithography (SLA) technology, digital light processing (DLP), and fused deposition modeling (FDM). These became more accessible for personal manufacturing, mainly because of economic reasons. On the other hand, researchers started to explore new acoustic instrument designs with advanced printing technologies like Binder Jetting or PolyJet [10]. These machines differ in material choices, resolution, and how they apply the support materials, which are crucial in musical instrument acoustics. The use of new materials in printing to improve the acoustics provided opportunities to decrease the chamber's size while preserving the loudness of the instrument based on cell structured assembly [11]. The 3D printed flute [10] adopted an inject printing technology which could infuse materials with different properties simultaneously in a single build with higher resolution rates (Fig. 2). Despite the fact that the PolyJet technology improved some of the limitations FDM posed like cracking between layers, high tolerance, lower resolution, and non-airtight walls, it still cannot eliminate the material decomposition problem fully.



Figure 2. The 3D printed flute manufactured with the PolyJet technology [10].

For these reasons, PolyJet offers a great technology for prototyping, yet not for manufacturing [10]. This led the researchers to direct efforts in manufacturing new forms of instrument parts such as valves, mouthpieces, or unconventional tonal series [10, 12–16]. The motivation behind these research work not only demands overcoming the limitation of the current manufacturing technologies to create complete working instruments, but also it encourages instrument designers and researchers to experiment creative, artistic interaction methods, explore customized tuning in addition to new solutions for improved ergonomics, acoustics, and aesthetics.

At the current state, despite the availability of new fabrication tools, acoustic instrument production still requires great human effort in modeling, pre, and post editing. In

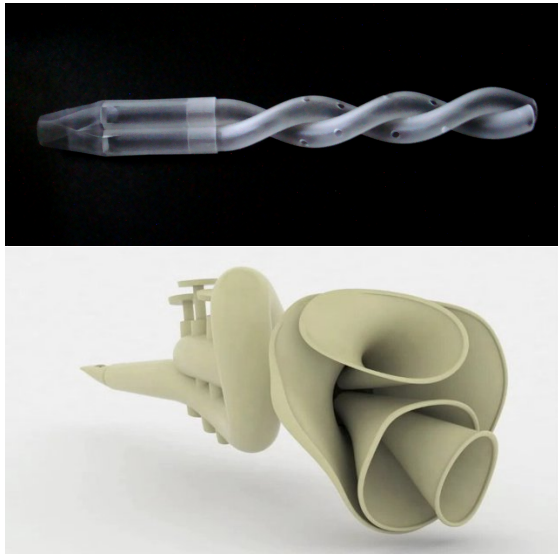


Figure 3. 3D printed double helix flute on top [13], a trumpet with multiple tubes of different radii on the bottom [10].

order to enable analyzing the digital models of instruments before materializing them, researchers developed modal profiles computed with the Finite Element Method (FEM) from 3D models of instruments [17, 18]. They proposed a new modal modeling method integrating FEM analysis into the open-source Computer-Aided Design (CAD) environment for computational estimation of the fabricated targets.

Overall, additive manufacturing was not restricted with the acoustic instrument design but also it was extended in the design of professional uses of hybrid and augmented instruments. Some researchers manufactured instrument bodies and used 3D printing with aesthetic concerns in mind [19]. Those projects not solely leveraged the flexibility or the ability of rapid prototyping, but also proved that AM can provide new forms of aesthetics. For example, Michon et al. used these techniques to design hybrid and augmented mobile instruments [20, 21]. Artists, researchers, and designers leveraged the availability of AM tools to customize their instruments in various ways. It also becomes an integral part of the design for bodily interactions in artistic performances [3].

### 2.1.2 Subtractive Manufacturing

Subtractive manufacturing is the process by which 3D objects are constructed by successively cutting material away from a solid block of material [22]. Although SM tools such as laser cutters, vinyl cutters, or CNC milling machines, are still an important part of the instrument design process, as opposed to additive manufacturing, these processes have less common use among the new musical interface researchers and designers. This is mainly because of the lack of availability of SM tools in music research labs or their requirement of technical knowledge of machinery use. The existing examples are mostly guitar designs manufactured with wood CNC machines and aimed to improve

acoustic characteristics of the instrument by high accuracy of the tools [23].

Additionally, some examples of instruments created in this realm prove that the application of this manufacturing method is not restricted with production purposes but also concern aesthetics and customizable interactions. A commercial carbon fiber guitar was designed and fabricated using milling machines [24] (Fig. 4). *Pipeline*, a brass pan flute with customized tuning, was manufactured using a combination of subtractive manufacturing techniques including rotary milling, turning, and laser cutting [25] (Fig. 5). Subtractive processes, in contrast to 3D printing, do not always offer flexibility; rather, they constrain design freedom due to the need for fixtures, diverse tooling, and the difficulty of the cutter in reaching deeper locations when fabricating complex geometries [26, 27]. For example, the initial design of the *Pipeline* had to be modified due to the dimensions and fixture limitations of the rotary table.

In general, SM becomes a priority when metal pieces need to be processed. Although the innovations in additive manufacturing started to offer 3D printing metal materials, in many cases, CNC post-machining is required to create fine features such as threads, to ensure functionality, and for surface finish [28]. The combination suggests an emerging and efficient fabrication method; hybrid manufacturing which is discussed in Section 2.3.



Figure 4. The carbon fiber acoustic guitar manufactured by using advanced composites and machining techniques [24].

## 2.2 Industrial Manufacturing

Some of the industrial manufacturing processes such as injection molding are used to build instrument parts or bodies. This technique provided the flexibility in designing forms and allowed musicians to experiment with new materials in their design and. It offers an efficient tool for rapid prototyping of multiple identical parts. Because the creation of industrial manufacturing tools (jigs, molds,...) are expensive and time-consuming, these processes are less common as personal manufacturing tools [3]. Ted Brewer's violin is an example of manufacturing the instrument body for his electric violin using injection molding [29]. Other examples are Weinberg and Aimi's Beat-bugs [30], which were cast in clear urethane from rubber molds. These examples of injection molding focused on designing instrument bodies for electric instruments rather than acoustic instruments. Rautia and Koivurova reported that "Non-enforced plastics commonly used in injection molding of the body of an electric guitar are acoustically not as good



Figure 5. Pipeline: brass pan flute with customized tuning [25].

as wood” [31]. Again these instruments are results of musicians or designers collaborating with the manufacturers from the industry rather than personal fabrications. An interesting example of an iterative prototyping method bridging between the industrial manufacturing and personal fabrication could be Kalo and Essl’s approach of fabricating cymbals using incremental robotic sheet forming [32].

Similarly, acoustuments provided a design method which made the manufacturing possible by combining 3D printing with injection molding. The passive acoustically driven handheld devices are iterated by 3D printing for rapid prototyping purposes and suggested that the manufacturing of these toys can be extended to injection-molding for high-volume, low-cost fabrication [21].

### 2.3 Hybrid Manufacturing

The term of hybrid manufacturing refers to the fabrication methods and technologies which combine different processes including additive, subtractive, or other (joining, dividing, transformative, ...) manufacturing processes on the same machine. The main advantage of this emerging technology is that it offers freedom of additive manufacturing while retaining the precision and surface finish quality of CNC, as well as reducing the dependence on a single process. This freedom unlocks the limitation of 3D printed instrument fabrication using multi-tasking machines. There are some examples of automating existing manual fabrication methods for musical applications. For example, Kalo and Essl used incremental robotic sheet metal forming to form cymbals [32]. Their contribution complements commonly used methods like 3D printing; yet, hybrid manufacturing, as a newly growing field, still lacks examples of musical instruments manufactured this way.

Hybrid manufacturing is progressively becoming common with the use of machinable materials for 3D printing, mainly for purposes like decreasing tolerance and speeding up processes. New machines which can apply both AM and SM on the same run offer better prototyping opportunities for musical instrument design research. Additionally, the fine features and surface finishes of instruments can easily be achieved with hybrid approaches. Yet, it is likely

that it will take time for these tools to find their place in the research labs [22].

### 2.4 Digital Manufacturing & Electronics

An integral part of the new interface design is the electronic construction of digital and electric musical instruments. In addition to reliable mechanical systems that new manufacturing techniques allow designers to realize, the electronic construction has a big influence on robustness, functionality, reliability, and durability of instruments [2, 3]. For stable and robust electronic circuits, designers use printed circuit boards (PCB). This technology is getting extended to printable electronics with new advancements in manufacturing of numerous sensing mechanisms [33]. The printable electronics offer easier constructions for embedded and wearable musical instruments. An earlier work of a wearable musical instrument which used additive and PCB manufacturing is Hattwick and Malloch’s prosthetic instruments [3]. As in similar wearable musical interfaces, the printed circuit technology open us new opportunities for end products in professional and artistic use since it provides low-cost, easy-to-use electronic circuits manufacturing. It further extends the tools to print flexible electronic components and sensors which are light-weight, ultra-thin, stretchable, bendable, and easy to operate in high mobility applications [34].



Figure 6. The fabric keyboard is built with multi-layer textile sensors machine-sewn in a keyboard pattern [35].

Subsequently, the manufacturing tools for flexible electronics led the designers to build improved designs of wearable instruments. Wicaksono and Paradiso explored a multi-modal, fabric-based, stretchable keyboard for physical interaction based on “deformable musical interface” which detects different stimuli such as touch, pressure, stretch, proximity, and electric field [35] (Fig. 6). Similar to [35], researchers in the interaction design community built several multi-touch textile sensors for music performances [36, 37]. This manufacturing type could be one of the most available manufacturing methods, after additive manufacturing, which could free instrument designers and builders to depend on a particular manufacturer.

Freed reports that the difficulties faced during the experience of building Wessel’s Slabs became a motivation to



adopt new technologies that resulted in new designs and new materials for piezoresistive pressure and position sensing surfaces [38]. Wessel's prediction about the printable electronics with Inkjet technology [33, 34] in musical instrument design is nowadays advanced to producing flexible tactile sensor using additive manufacturing techniques [39].

AM with embedded electronic components in the print offers a new manufacturing method in prototyping electronic circuits. In the near future, 3D printing electronics can offer a cost-effective and scalable fabrication technique as an alternative to conventional fabrication methods, most of which are complex, expensive and time-consuming [39]. Hybrid AM processes provide not only improvements for the form and appearance of final products, but also for electronics functionality with embedding most commonly used digital elements of DMIs (passive sensors, accelerometers). Fig. 7 gives a simple example designed using CAD modeling tools and prototyped with AM process [40].



Figure 7. Gaming dice with electronic circuit mechanically designed into substrate. It consists of a micro-controller, MEMS accelerometer, batteries and LEDs [40].

Whereas there has not been an example of this technology in the musical instrument design, not to the extent of author's knowledge, the growing interest in designing digital musical instruments and interfaces with 3D printing tools creates room to incorporate the electronic construction in the existing manufacturing processes. As a new technology, 3D printing electronics currently has limitations like complexity, time demand, higher cost. Hopefully, with more cost-effective machines, manufacturing light-weight, and compact flexible/stretchable electronics in fabrication labs will not only help to develop new interaction methods but also encourage designers to develop wearable interfaces for intermedia performances like combinations of music, dance, and theater.

### 3. FUTURE APPLICATIONS

Musical instrument design requires a lot of hands-on study in the fabrication labs (FabLabs). As the manufacturing tools have become affordable, higher number of fabrication labs have been founded in the academic makerspaces, and universities and research institutions have begun to offer more musical acoustics and instrument design classes [41–44]. Access to the manufacturing tools, specifically to 3D printers and laser cutters, brings fabrication processes to the classroom not only for university-level students but also for the primary school students. Harriman emphasizes the importance of digital musical instrument design in children's education [45], and Eisenberg discusses the challenges in the way of incorporating digital manufacturing into the classroom for children [46]. On the other side, researchers direct their efforts to design portable digital manufacturing tools to reach less accessible parts of the world [47].

One of the changes occurring in the digital fabrication area is in hybrid manufacturing. Although as of now there are some examples of hybrid manufacturing, there are a lot of improvements needed in this area, specifically for more cost-effective tools. Musical instrument design could benefit this technique in various ways by combining most commonly used techniques either for fast prototyping or for end products. These tools, in addition to portable digital manufacturing tools, can also enhance personal fabrication and change the modalities of industry-academia collaborations. Hybrid approaches are not limited to combinations of manufacturing tools; virtual reality (VR) is becoming one of the main tools that researchers merge into fabrication education and cloud control. Researchers use VR simulations for remote control or teaching purposes which accelerates the need for tactile sensors and haptic feedback mechanisms in VR tools generally. On the other hand, fast prototyping opportunities, which come with advanced digital manufacturing tools, propose customizable controllers for VR to overcome the human sensory limitations. The interaction between the two fields can be beneficial for both areas in creating interactive tools and opening up new opportunities.

While manufacturing and material science provide new ways of fabricating instruments, researchers are finding promising results with acoustic modeling of fabricated instruments. With improved computation of 3D data, 3D scanners can be used in modal analysis of manufactured instrument by reverse CAD modeling. Unfortunately, the current state of the professional 3D scanners still poses challenges in obtaining accurate models. The technology requires post-processing of the scanner data before FEM simulation.

### 4. CONCLUSIONS

While the design capabilities with the fabrication methods discussed in this paper depend upon the financial and institutional infrastructures, digital manufacturing tools are becoming available with a greater variety and frequently used in music research and education. This paper discusses



the current digital fabrication techniques used in musical instrument design. It presents an overview of what future directions are available and how they can be applied to musical instrument making and artistic interaction design.

The innovations creating new opportunities for musical expression are not limited solely to the manufacturing processes. New materials can provide better acoustics or more playable and robust musical instruments. The research conducted in this area such as print composites of wood and polyester, or bio-composites are not covered in this paper.

## 5. REFERENCES

- [1] H. Cather, R. D. Morris, M. Philip, and C. Rose, *Design engineering*. Elsevier, 2001.
- [2] I. Hattwick and M. M. Wanderley, "Design of hardware systems for professional artistic applications," in *Proceedings of the 17th International Conference on New Interfaces for Musical Expression (NIME-17)*, 2017.
- [3] I. Hattwick, J. Malloch, and M. M. Wanderley, "Forming shapes to bodies: Design for manufacturing in the prosthetic instruments," in *Proceedings of the 14th International Conference on New Interfaces for Musical Expression (NIME-14)*, 2014, pp. 443–448.
- [4] C. K. Chua and K. F. Leong, *3D Printing and Additive Manufacturing: Principles and Applications (with Companion Media Pack) of Rapid Prototyping Fourth Edition*. World Scientific Publishing Company, 2014.
- [5] Hovalabs. Hovalin website. [Online]. Available: <http://www.hovalabs.com/hova-instruments/hovalin>
- [6] S. Summit, "System and method for designing and fabricating string instruments," Dec. 13 2016, uS Patent 9,519,733.
- [7] O. P. Perry, David. Modular fiddle. [Online]. Available: <https://openfabpdx.com/modular-fiddle/>
- [8] B. Chan. Designing a 3D Printed Acoustic Violin, Formlabs. [Online]. Available: <https://formlabs.com/blog/designing-a-3d-printed-acoustic-violin/>
- [9] J. Walter-Herrmann and C. Büching, *FabLab: Of machines, makers and inventors*. transcript Verlag, 2014.
- [10] A. Zoran, "The 3D printed flute: digital fabrication and design of musical instruments," *Journal of New Music Research*, vol. 40, no. 4, pp. 379–387, 2011.
- [11] A. Zoran and P. Maes, "Considering virtual & physical aspects in acoustic guitar design," in *Proceedings of the 8th International Conference on New Interfaces for Musical Expression (NIME-08)*, 2008, pp. 67–70.
- [12] S. Leitman and J. Granzow, "Music maker: 3D printing and acoustics curriculum," in *Proceedings of the 16th Conference on New Interfaces for Musical Expression (NIME-16)*. Brisbane, Australia, 2016.
- [13] M. Dabin, T. Narushima, S. T. Beirne, C. H. Ritz, and K. Grady, "3D Modelling and Printing of Microtonal Flutes," in *Proceedings of the 16th International Conference on New Interfaces for Musical Expression (NIME-16)*, Brisbane, Australia, 2016, pp. 11–15.
- [14] N. J. Bailey, T. Cremel, and A. South, "Using acoustic modelling to design and print a microtonal clarinet," in *Proceedings of the 9th Conference on Interdisciplinary Musicology (CIM14)*, Berlin, Germany, 2014, pp. 4–6.
- [15] J. Granzow, "Additive manufacturing for musical applications," Ph.D. dissertation, Ph.D. Thesis, Stanford University, Stanford, CA, USA, 2017.
- [16] V. Lorenzoni, E. Doubrovski, and J. Verlinden, "Embracing the digital in instrument making: Towards a musician-tailored mouthpiece by 3D printing," in *Proceedings of the Stockholm Music Acoustics Conference 2013, SMAC 2013, Stockholm (Sweden), 30 July-3 August, 2013*, 2013.
- [17] R. Michon, C. Chafe, and J. Granzow, "3D Printing and Physical Modeling of Musical Instruments: Casting the Net," in *Proceedings of Int. Conf. Sound and Music Computing (SMC-18)*, 2018.
- [18] N. Plath, "3D Imaging of Musical Instruments: Methods and Applications," in *Computational Phonogram Archiving*. Springer, 2019, pp. 321–334.
- [19] E. Goldemberg and V. Zalcberg. Monad studio. [Online]. Available: <https://www.monadstudio.com/About-MONAD-Studio>
- [20] R. Michon, J. O. Smith, M. Wright, C. Chafe, J. Granzow, and G. Wang, "Mobile music, sensors, physical modeling, and digital fabrication: Articulating the augmented mobile instrument," *Applied Sciences*, vol. 7, no. 12, p. 1311, 2017.
- [21] G. Laput, E. Brockmeyer, S. E. Hudson, and C. Harrison, "Acoustruments: Passive, acoustically-driven, interactive controls for handheld devices," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2015, pp. 2161–2170.
- [22] Z. Zhu, V. G. Dhokia, A. Nassehi, and S. T. Newman, "A review of hybrid manufacturing processes—state of the art and future perspectives," *International Journal of Computer Integrated Manufacturing*, vol. 26, no. 7, pp. 596–615, 2013.
- [23] A. Krimpenis and M. Chrysikos, "3D parametric design and CNC manufacturing of custom solid wood electric guitars using CAD/CAM technology," *Wood Material Science & Engineering*, pp. 1–15, 2017.
- [24] B. Guitars, "Blackbird rider acoustic," 2008.
- [25] D. Cavdir, "Pipeline: custom tuned brass pan flute," 2018. [Online]. Available: <http://www.dogacavdir.com/pipeline-si5x>

- [26] K. Karunakaran, A. Bernard, S. Suryakumar, L. Dembinski, and G. Taillandier, "Rapid manufacturing of metallic objects," *Rapid Prototyping Journal*, vol. 18, no. 4, pp. 264–280, 2012.
- [27] W. Gao, Y. Zhang, D. Ramanujan, K. Ramani, Y. Chen, C. B. Williams, C. C. Wang, Y. C. Shin, S. Zhang, and P. D. Zavattieri, "The status, challenges, and future of additive manufacturing in engineering," *Computer-Aided Design*, vol. 69, pp. 65–89, 2015.
- [28] W. E. Frazier, "Metal additive manufacturing: a review," *Journal of Materials Engineering and Performance*, vol. 23, no. 6, pp. 1917–1928, 2014.
- [29] K. R. Zappas, "The science of sound: Examining the role of materials in musical instruments," *JOM*, vol. 59, no. 8, p. 13, 2007.
- [30] G. Weinberg, "The beatbug—evolution of a musical controller," *Digital Creativity*, vol. 19, no. 1, pp. 3–18, 2008.
- [31] V. Rautia and H. Koivurova, "Method for manufacturing musical instrument and a musical instrument," May 8 2007, uS Patent 7,214,866.
- [32] A. Kalo and G. Essl, "Individual fabrication of cymbals using incremental robotic sheet forming," in *Proceedings of the 16th International Conference on New Interfaces for Musical Expression (NIME-16)*, 2016, pp. 287–292.
- [33] M. Mantysalo, V. Pekkanen, K. Kaija, J. Niittyinen, S. Koskinen, E. Halonen, P. Mansikkamaki, and O. Hameenoja, "Capability of inkjet technology in electronics manufacturing," in *Electronic Components and Technology Conference, 2009. ECTC 2009. 59th. IEEE*, 2009, pp. 1330–1336.
- [34] Z. Yin, Y. Huang, N. Bu, X. Wang, and Y. Xiong, "Inkjet printing for flexible electronics: Materials, processes and equipments," *Chinese Science Bulletin*, vol. 55, no. 30, pp. 3383–3407, 2010.
- [35] I. Wicaksono and J. A. Paradiso, "Fabrickeyboard: Multimodal textile sensate media as an expressive and deformable musical interface," *Proceedings of the 17th International Conference on New Interfaces for Musical Expression (NIME-17)*, Aalborg, Denmark, 2017.
- [36] J.-S. Roh, Y. Mann, A. Freed, D. Wessel, U. Berkeley, A. Freed, A. Street, and D. Wessel, "Robust and reliable fabric, piezoresistive multitouch sensing surfaces for musical controllers," in *Proceedings of the 11th International Conference on New Interfaces for Musical Expression (NIME-11)*, 2011, pp. 393–398.
- [37] M. Donneaud, C. Honnet, and P. Strohmeier, "Designing a multi-touch etextile for music performances," in *Proceedings of the 17th International Conference on New Interfaces for Musical Expression (NIME-17)*, Aalborg, Denmark, 2017, pp. 15–19.
- [38] A. Freed, "David wessels slabs: a case study in preventative digital musical instrument conservation," *Proceedings of Int. Conf. Sound and Music Computing, (SMC-16)*, 2016.
- [39] C. Liu, N. Huang, F. Xu, J. Tong, Z. Chen, X. Gui, Y. Fu, and C. Lao, "3D printing technologies for flexible tactile sensors toward wearable electronics and electronic skin," *Polymers*, vol. 10, no. 6, p. 629, 2018.
- [40] E. Macdonald, R. Salas, D. Espalin, M. Perez, E. Aguilera, D. Muse, and R. B. Wicker, "3D printing for the rapid prototyping of structural electronics," *IEEE access*, vol. 2, pp. 234–242, 2014.
- [41] V. Wilczynski, "Academic maker spaces and engineering design," in *American Society for Engineering Education*, vol. 26, 2015, p. 1.
- [42] T. Bertrand, K. Kaczmarek, and L. Wilen, "Musical acoustics & instrument design: When engineering meets music," in *ICMC*, 2015.
- [43] N. Villicaña-Shaw, S. Salazar, and A. Kapur, *The Machine Lab: A Modern Classroom to Teach Mechatronic Music*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 2017.
- [44] A. Zoran, "Hybrid craft: showcase of physical and digital integration of design and craft skills," in *ACM SIGGRAPH Art Gallery*. ACM, 2015, pp. 384–398.
- [45] J. Harriman, "Start'em young: digital music instrument for education," in *Proceedings of the 15th International Conference on New Interfaces for Musical Expression (NIME-15)*, 2015, pp. 70–73.
- [46] M. Eisenberg, "3D printing for children: What to build next?" *International Journal of Child-Computer Interaction*, vol. 1, no. 1, pp. 7–13, 2013.
- [47] N. Peek and I. Moyer, "Popfab: A case for portable digital fabrication," in *Proceedings of the Eleventh International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, 2017, pp. 325–329.

# Real Time Audio Digital Signal Processing With Faust and the Teensy

Romain Michon,<sup>1,2</sup> Yann Orlarey,<sup>1</sup> Stéphane Letz,<sup>1</sup> and Dominique Fober<sup>1</sup>

<sup>1</sup> GRAME-CNCMC, 11 Cours de Verdun-Gensoul, 69002 Lyon (France)

<sup>2</sup> Center for Computer Research in Music and Acoustics, 660 Lomita Ct., Stanford CA 94350-8180 (USA)  
rmichon@ccrma.stanford.edu

## ABSTRACT

This paper introduces a series of tools to program the Teensy development board series with the FAUST programming language. `faust2teensy` is a command line application that can be used both to generate new objects for the Teensy Audio Library and standalone Teensy programs. We also demonstrate how `faust2api` can produce Digital Signal Processing engines (with potential polyphony support) for the Teensy. Details about the implementation and optimizations of these systems are provided and the results of various tests (i.e., computational, latency, etc.) are presented. Finally, future directions for this work are discussed through a discussion on bare-metal implementation of real-time audio signal processing applications.

## 1. INTRODUCTION

Arduinos<sup>1</sup> contributed to the spreading of microcontrollers by making them more accessible through a high level programming language (which is essentially a subset of C++), various domain-specific libraries, and an Integrated Development Environment (IDE) allowing to export the generated firmware to the board using USB.

The impact of the “Arduino revolution” on the computer music/NIME (New Interfaces for Musical Expression) community has been significant and gave birth to hundreds of new music controllers and instruments.

In parallel of that, the rise of embedded Linux platforms with their potential applications to real-time audio signal processing also impacted the way we approach digital lutherie [1]. A series of tools (both hardware and software) such as Satellite CCRMA [2] and the BELA<sup>2</sup> [3] (to only name a few) have been exploiting the potential of these new technologies. The BELA is especially interesting to us as it adds audio-rate analog I/Os and low-latency audio processing capabilities to the BeagleBone Black.<sup>3</sup> More specific applications and experiments have also been

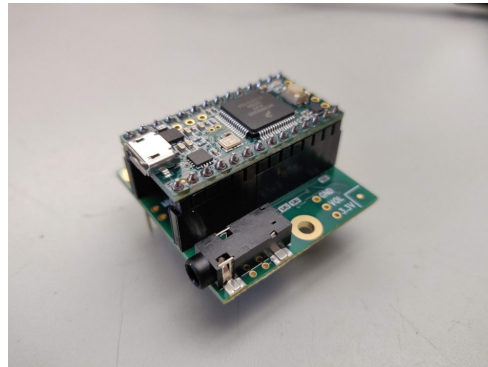


Figure 1. The Teensy 3.2 and its audio shield.

targeting specialized boards such as FPGAs (Field Programmable Gate Array) (e.g., Digilent Zybo Series<sup>4</sup>) [4,5] and DSPs (Digital Signal Processing) (e.g., Analog Devices SHARC Audio Module,<sup>5</sup> etc.). The main drawback of these platforms is their price: fully functional “all-in-one” solutions can’t be found for less than 100USD.

On the other hand, recent microcontrollers are cheap, offer an heightened computational power, and can be used to synthesize/process audio signals. Some of them such as the ARM Cortex-M4F<sup>6</sup> even include a dedicated Floating-Point Unit (FPU) and support for DSP instructions, making them a well-suited platform for sound synthesis/processing.

ARM Cortex-M4 microcontrollers are used at the heart of PJRC’s Teensy<sup>7</sup> development board series whose price averages 25USD. Since the Cortex-M4 hosts its own DAC,<sup>8</sup> sound can be synthesized and played directly from the Teensy. PJRC also offers an audio shield for the Teensy that essentially upgrades it with a SGTL5000 Audio Codec providing a 16bits 44.1kHz stereo audio input and output (see Figure 1) for 15USD!

The Teensy comes with an Audio Library<sup>9</sup> where basic DSP objects (e.g., oscillators, effects, Karplus-Strong, etc.) implemented in C++ can be patched with a high level API. This task is facilitated by an online graphical environ-

<sup>1</sup> <https://www.arduino.cc/>. All URLs presented in this paper were verified on Feb. 4, 2019.

<sup>2</sup> <https://bela.io/>

<sup>3</sup> <https://beagleboard.org/bone>

<sup>4</sup> <https://store.digilentinc.com/zybo-z7-zynq-7000-arm-fpga-soc-development-board/>

<sup>5</sup> <https://wiki.analog.com/resources/tools-software/sharc-audio-module>

<sup>6</sup> <https://developer.arm.com/products/processors/cortex-m/cortex-m4>

<sup>7</sup> <https://www.pjrc.com/teensy/>

<sup>8</sup> DigitaltoAudioConverter

<sup>9</sup> [https://www.pjrc.com/teensy/td\\_libs\\_Audio.html](https://www.pjrc.com/teensy/td_libs_Audio.html)

ment<sup>10</sup> where objects can be connected by drawing patch chords between them.

The standard DSP objects of the Teensy Audio Library are relatively basic. New objects can be implemented in C++, which is often out of reach to people in the DIY (Do It Yourself) community who might use the Teensy. Also, since not all the microcontrollers used in the Teensy series have an FPU, the standard DSP objects of the Teensy Audio Library are all implemented in fixed point.

FAUST [6] is a functional programming language for efficient real-time audio signal processing. The FAUST compiler can generate DSP code/classes in various languages (i.e., C, C++, Java, JS, LLVM, WebAssembly, etc.) from a given FAUST program. The FAUST DSP libraries implement hundreds of algorithms for sound synthesis and processing.

In this paper, we introduce a series of tools to program the Teensy with FAUST.<sup>11</sup> First, we present `faust2teensy`, a command-line application to generate new DSP objects for the Teensy Audio Library. Then, we introduce a new target for `faust2api` [7] allowing us to generate DSP engines (with potential polyphony support) for the Teensy. We also demonstrate how ready-to-use Teensy programs can be written in FAUST. In that case, the parameters of a FAUST program (e.g., the frequency of an oscillator, etc.) can be directly mapped to the analog and digital inputs of the Teensy. Finally, we evaluate the performances of these systems and we present future directions for this type of work.

## 2. CREATING NEW OBJECTS FOR THE TEENSY AUDIO LIBRARY

### 2.1 Generating DSP Objects With `faust2teensy`

`faust2teensy` is a command-line tool that can be used to generate new DSP objects compatible with the Teensy Audio Library.<sup>12</sup> For this, the `-lib` option must be provided when calling `faust2teensy`:

```
faust2teensy -lib MyFaustSynth.dsp
```

which will yield a package containing two C++ files: `MyFaustSynth.h` and `MyFaustSynth.cpp` (see Figure 2) implementing a class called `MyFaustSynth`.

These files can either be placed in the source of the Teensy Audio Library or in their own library. In both cases, the `.h` file should be included at the beginning of the Teensy program and then called and connected at least to a DAC (Digital to Audio Converter) just like any other object of the Teensy Audio Library:

```
#include <Audio.h>
#include <MyFaustSynth.h>
MyFaustSynth myFaustSynth;
AudioOutputAnalog dac;
```

<sup>10</sup> <http://www.pjrc.com/teensy/gui/index.html>

<sup>11</sup> All the tools presented in this paper are open source and have been integrated to the FAUST distribution which can be found on GitHub: <https://github.com/grame-cncm/faust>.

<sup>12</sup> We'll see in §4 that `faust2teensy` can also be used to generate ready-to-use Teensy programs.

```
AudioConnection
  patchCord0(myFaustSynth,dac);
void setup() {
  AudioMemory(2);
}
void loop() {
}
```

Listing 1. Simple Teensy program using a FAUST-generated DSP object with the built-in DAC of the Teensy.

`AudioOutputAnalog` corresponds to the built-in DAC of the Teensy but `AudioOutputI2S` could be used instead, in case the Teensy is equipped with an audio shield. In that case, an `AudioControlSGTL5000` object should also be instantiated and multi-channel audio connections can be used:

```
#include <Audio.h>
#include <MyFaustSynth.h>
MyFaustSynth myFaustSynth;
AudioOutputI2S dac;
AudioControlSGTL5000 audioShield;
AudioConnection
  patchCord0(myFaustSynth,0,dac,0);
AudioConnection
  patchCord1(myFaustSynth,0,dac,1);
void setup() {
  AudioMemory(2);
  audioShield.enable();
}
void loop() {
}
```

Listing 2. Simple Teensy program using a FAUST-generated DSP object with the Teensy Audio Shield.

Note that the number of audio inputs and outputs of the generated object depends on the FAUST program, hence `MyFaustSynth` could have more than one output, in which case the wiring of the audio connections could look like:

```
AudioConnection
  patchCord0(myFaustSynth,0,dac,0);
AudioConnection
  patchCord1(myFaustSynth,1,dac,1);
```

The following Code Listing presents an example FAUST program implementing a band-limited sawtooth wave oscillator that could be used as `MyFaustSynth.dsp`:

```
import("stdfaust.lib");
freq = nentry("f",300,50,2000,0.01) :
  si.smoo;
gain = nentry("g",0.5,0,1,0.01) :
  si.smoo;
process = os.sawtooth(freq)*gain;
```

Listing 3. FAUST program implementing a band-limited sawtooth oscillator controllable with some UI elements.

`f` controls the frequency of the oscillator and `g` its gain. Both are processed by `si.smoo` which exponentially interpolates samples, preventing discontinuities.



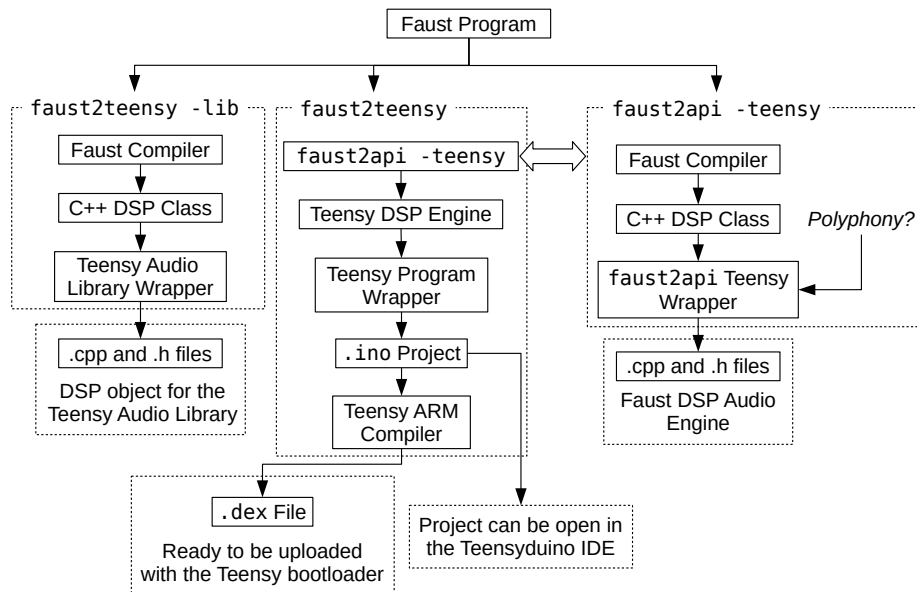


Figure 2. Overview of the various tools to use FAUST on the Teensy.

The value of  $f$  and  $g$  can be set at any point by calling the `setParamValue` method which takes the name of the FAUST parameter and its corresponding value as arguments. Note that this method can also be used with `faust2api` (see §3) as it is inherited from the same parent class (`MapUI`) of the FAUST architectures system. Hence, the frequency of the FAUST sawtooth oscillator presented in Code Listing 3 could be set randomly every two-hundred milliseconds by modifying the implementation of the `loop` function<sup>13</sup> of Code Listing 1:

```
void loop() {
  myFaustSynth.setParamValue("f",
    random(50,2000));
  delay(200);
}
```

## 2.2 Implementation

`faust2teensy` is a simple bash script calling the command-line FAUST compiler to generate the C++ DSP class corresponding to a given FAUST program (see Figure 2). The generated C++ code is pasted into a wrapper C++ file (also called architecture in the FAUST world) implementing a generic object for the Teensy Audio Library. This file and its corresponding header file are then packaged in a zip file which is provided to the user.

<sup>13</sup> `loop` is a standard Arduino function that is repeated until the device running it is powered off. In other words, it corresponds to the main thread of the system. `delay` is also an Arduino function that can be used to pause the thread for a given duration in milliseconds.

## 3. GENERATING FAUST DSP ENGINES FOR THE TEENSY

### 3.1 Monophonic DSP Engine

FAUST can also be used to generate ready-to-use DSP engines for the Teensy. In that case, the strategy consists in letting FAUST taking care entirely of the sound synthesis/processing portion of the program.

The FAUST program presented in Code Listing 3 can be turned into a DSP engine using `faust2api` by running the following command line in a terminal:

```
faust2api -teensy MyFaustSynth.dsp
```

Just like `faust2teensy` (see §2), the generated package contains a `.cpp` and a `.h` file that can be called directly in a Teensy program:

```
#include <MyFaustSynth.h>
int SR = 44100; // Sampling Rate
int BS = 128; // Block Size
MyFaustSynth myFaustSynth(SR,BS);
void setup() {
  myFaustSynth.setDevice(0,0);
  myFaustSynth.start();
}
void loop() {
  myFaustSynth.setParamValue(
    "f",random(50,2000));
  delay(200);
}
```

Listing 4. Teensy program using a FAUST-generated monophonic DSP engine.

Here, the `MyFaustSynth` object relies on the Teensy Audio Library which it calls internally. The `setDevice` method allows the programmer to specify the hardware

Device ID	Device Description
0	Teensy Audio Shield
1	Teensy Audio Shield (Quad)
2	Built-In ADC
3	Built-In ADC (Stereo)
4	Teensy Audio Shield (Slave Mode)
5	Pulse Density Modulated Bitstream
6	Time Division Multiplexed Frame
7	USB: receive stereo audio from computer

Table 1. Input devices ID for the `setDevice` method (directly taken from the Teensy Audio Library).

Device ID	Device Description
0	Teensy Audio Shield
1	Teensy Audio Shield (Quad)
2	SPDIF
3	PT8211 DAC
4	Built-In DAC
5	Built-In DAC (Stereo)
6	PWM
7	Teensy Audio Shield (Slave Mode)
8	Time Division Multiplexed Frame
9	ADAT
10	USB: send stereo audio to computer

Table 2. Output devices ID for the `setDevice` method (directly taken from the Teensy Audio Library).

input (see Table 1) and output (see Table 2) of the DSP engine. `start` launches computation and `setParameterValue` is used to change the value of a specific parameter (as with `faust2teensy`). All the other standard `faust2api` methods [7] are also available.

### 3.2 Polyphonic DSP Engine

While the benefits of using a Teensy monophonic DSP engine generated with `faust2api` over `faust2teensy` might be questionable, `faust2api` also allows us to generate polyphonic audio engines that can be used with a specific API.

A FAUST program can be made “polyphony-compatible” simply by declaring the `freq`, `gain`, and `gate` parameters.<sup>14</sup> In that case, an audio effect common to all voices can be declared using the `effect` standard declaration. Hence, Code Listing 3 can be easily modified to make it polyphony-compatible (`MyFaustSynthPoly.dsp`):

```
import("stdfaust.lib");
freq = nentry("freq",300,50,2000,0.01);
gain = nentry("gain",0.5,0,1,0.01);
gate = button("gate");
envelope = en.asr(0.01,gain,0.01,gate);
process = os.sawtooth(freq)*envelope;
```

<sup>14</sup> <https://faust.grame.fr/doc/manual/index.html#midi-polyphony-support>

```
effect = +~@(ma.SR*0.15)*0.3; // echo
```

Listing 5. FAUST program implementing a MIDI-controllable polyphonic synthesizer.

The FAUST program presented in Code Listing 5 can be turned into a polyphonic DSP engine by running the following command line:

```
faust2api -teensy -nvoices 4 -effect auto
MyFaustSynthPoly.dsp
```

Note that `-nvoices` allows us to specify the maximum number of voices of polyphony of the engine and that `-effect auto` connects all the voices to the effect declared in the `effect` standard declaration.

The generated DSP engine allows for the use of polyphony-related methods [7] such as `keyOn`, `keyOff`, `newVoice`, `deleteVoice`, `setVoiceParamValue`, etc. Code Listing 6 demonstrates the use of a polyphonic DSP engine by generating random major chords.

```
#include <MyFaustSynthPoly.h>
MyFaustSynthPoly myFaustSynth(44100,128);
void setup() {
  myFaustSynth.setDevice(0,0);
  myFaustSynth.start();
}
void loop() {
  int root = random(40,80);
  int M3 = root+4; int P5 = root+7;
  myFaustSynth.keyOn(root,100);
  myFaustSynth.keyOn(M3,100);
  myFaustSynth.keyOn(P5,100);
  delay(1000);
  myFaustSynth.keyOff(root,100);
  myFaustSynth.keyOff(M3,100);
  myFaustSynth.keyOff(P5,100);
  delay(500);
}
```

Listing 6. Teensy program using a FAUST-generated polyphonic DSP engine.

Figure 2 gives an overview of the implementation of this system.

## 4. USING FAUST TO PROGRAM THE TEENSY

`faust2teensy` can be used in “standalone mode” to fully program the Teensy directly from FAUST without writing a single line of Arduino code. This is done through the use of specific metadata in the declaration of the name of the parameters of the FAUST program. Hence, `[io: AN]` can be used to connect the `N` analog pin to the current parameter. The same approach is used for digital pins using the `[io: DN]` metadata.

Global metadata can be declared as well to configure the sampling rate (declare `SR`), the block size (declare `BS`), and the audio input and output (declare `device` – see Tables 1 and 2 for a list of available inputs and outputs) of the Teensy.

Code Listing 7 presents a FAUST program where analog pins 0 and 1 of the Teensy control respectively the fre-

quency and the gain of a sawtooth oscillator and digital pin 0 the fact that it's on or off.

```
declare SR "44100";
declare BS "128";
declare device "{0,0}";
import("stdfaust.lib");
f = nentry("f[i0: A0]",
  300,50,2000,0.01) : si.smoo;
g = nentry("g[i0: A1]",
  0.5,0,1,0.01) : si.smoo;
t = nentry("t[i0: D0]",
  0,0,1,1) : si.smoo;
process = os.sawtooth(f)*g*t;
```

Listing 7. Standalone FAUST Teensy program.

Note that the range of analog pins on the Teensy is automatically mapped to that of the corresponding FAUST parameter. Hence, in the case of the  $f$  parameter in Code Listing 7, if the Teensy uses 10 bits integers to store the values of the samples acquired by analog pin 0, 0 will correspond to a value of  $f$  of 50 and 1023 to a value of  $f$  of 2000. The same is true for digital pins.

`faust2teensy` can be used in standalone mode simply by running the following command in the terminal:

```
faust2teensy MyFaustSynth.dsp
```

In that case, `faust2teensy` will automatically call the Teensy bootloader to upload the generated firmware, so the Arduino IDE doesn't have to be used at all! Note that the `-vb` option can be added to verbose the output of the compilation process.

## 5. EVALUATION

The tools presented in §2-4 have been evaluated through the simple FAUST program presented in Code Listing 8 which implements a sawtooth oscillator from scratch.

```
import("stdfaust.lib");
freq = hslider("freq",400,50,2000,0.01);
frac(n) = n - floor(n);
sawtooth(f) = +(f/ma.SR)^frac : *(2)-1;
process = sawtooth(freq) <: _,_;
```

Listing 8. Standalone FAUST Teensy program.

This algorithm was chosen for its simplicity (only three additions/subtractions and one multiplication). The corresponding C++ code generated by the FAUST compiler was used with `faust2api` in polyphonic mode (see §3.2) to measure the number of cycles per seconds on the Teensy under various conditions. This code was re-written “by hand” to use fixed points instead of floating points without changing the algorithm (the FAUST compiler can only generate floating point DSP code).

Tests were carried out on a Teensy 3.2 (MK20DX256VLH7 Cortex-M4 72MHz) and a Teensy 3.6 (MK66FX1M0VMD18 Cortex-M4F 180MHz). Since 3.6 has an FPU, floating point instructions were forced by

using the following options during compilation: `-mfloat -abi=hard -mfpu=fpv4-sp-d16` (selects a hardware floating-point unit conforming to the single precision variant of the FPv4 architecture) when testing DSP code using floating points. Since 3.2 doesn't have an FPU, no specific C++ compilation options for floating points were selected (emulated floating points).

The results of our tests are presented in Table 3. All tests were carried out at a sampling rate of 44.1KHz. We chose 128 samples as our maximum test block size since using higher block sizes doesn't seem to impact computation. 8 samples is the smallest block size that we were able to use on the Teensy. “MaxPoly” corresponds to the maximum number of voices of polyphony based on Code Listing 8 that we were able to run.

As expected, the Teensy 3.6 outperforms the 3.2 for all tests. While there's only a gain factor of  $\sim 2.2$  between these two devices when using fixed point arithmetic, the 3.6 was 15 times more powerful in average than the 3.2 when using floating points. Hence, 63 parallel versions of the algorithm presented in Code Listing 8 (which corresponds to a total of 64 multiplications and 251 additions/-subtractions per sample) could be ran in parallel and added on the 3.6 while the 3.2 only allowed to play 4 voices. Another interesting element to note is that the impact of block size on computation is rather small. Hence, a block size of 8 samples is only 1.2 times more expensive in average than a block size of 128 samples (or greater) in most cases.

These performances could potentially be improved by using CMSIS-DSP instructions,<sup>15</sup> but since all the optimized math function of this library are vector-based, they're only useful for very specific kinds of algorithms. For instance, they would not help make the program presented in Code Listing 8 more efficient because of its internal feedback.

More complex algorithms such as the FAUST version of Zita-Verb (stereo feedback delay network) [8] were also ran successfully on the Teensy 3.6.

Finally, audio “round-trip” (analog to digital and then back to analog) latency measurements were carried out (also using a sampling rate of 44.1KHz). When using a block size of 128 samples, a latency of 10.5ms was measured. When using a block size of 8 samples, a latency of 1.2ms was measured!

## 6. FUTURE DIRECTIONS

The current set of metadata available to produce standalone Teensy programs with FAUST (see §4) is somewhat limited and could be easily extended. For example, it is currently not possible to map sensors using i2s (Integrated Inter-IC Sound Bus) to the parameters of a FAUST program, etc. We plan to expand the scope of these metadata in the future to allow users to program the Teensy in FAUST without making compromises.

Microcontrollers and CPUs for embedded systems now offer enough processing power to implement complex real-time audio signal processing algorithms, but little work has

<sup>15</sup> <http://www.keil.com/pack/doc/CMSIS/DSP/html>

	Block Size	Teensy 3.2	Teensy 3.6
int16	128	~2,004,700 c/s	~4,801,450 c/s
MaxPoly int16	128	34 (~32,150 c/s)	75 (~31,350 c/s)
int32	128	~1,113,700 c/s	~2,402,900 c/s
MaxPoly int32	128	20 (~23,950 c/s)	33 (~55,550 c/s)
float32	128	~222,650 c/s	~3,512,600 c/s
MaxPoly float32	128	4 (~25,700 c/s)	63 (~31,350 c/s)
int16	8	~1,774,000 c/s	~4,271,300 c/s
MaxPoly int16	8	30 (~49,050 c/s)	67 (~59,950 c/s)
int32	8	~986,100 c/s	~2,122,400 c/s
MaxPoly int32	8	17 (~43,200 c/s)	30 (~28,300 c/s)
float32	8	~196,000 c/s	~3,109,050 c/s
MaxPoly float32	8	3 (~70,150 c/s)	60 (~27,900 c/s)

Table 3. Number of cycles per second and maximum number of voices of polyphony for different data types, block sizes and Teensy boards based on the FAUST program presented in Code Listing 8.

been done towards bare-metal implementations on more advanced platforms. Indeed, while we don't think there's more work to be done on the Teensy side, we'd like to implement a series of tools similar to the ones presented in this paper targeting the Raspberry Pi (RPI).<sup>16</sup> The RPI 3 A+<sup>17</sup> only costs 25USD and is based on a Broadcom BCM2837B0 Cortex-A53 with 4 1.4GHz cores and 512MB of RAM. Beside the fact it offers much more processing power than the Teensy, the Cortex-A53 microarchitecture provides support for Neon,<sup>18</sup> which should allow further optimizations for floating points operations.

While the PI is not a microcontroller like the Teensy and therefore doesn't have any analog inputs for sensors, etc. it can be easily upgraded with an Analog to Digital Converter (ADC) such as an MCP3008<sup>19</sup> which costs less than 4USD. Similarly, the built-in audio codec of the PI is notorious to be low quality. The Fe-Pi Audio Z V2<sup>20</sup> is a sister board for the PI using the same SGTL5000 Audio Codec as the Teensy Audio Shield, and its cost is inferior to 12 USD. Hence, the total cost of this set-up is similar to the one of a Teensy 3.6 upgraded with an Audio Shield (~45USD) but provides much more processing power for potential bare-metal implementations.

## 7. CONCLUSIONS

Programming the Teensy for custom real-time audio signal processing applications is out of reach to most people in the DIY community. The tools presented in this paper provide a comprehensive way to carry out this type of task at a higher level using the FAUST programming language. Programmers benefit from hundreds of existing DSP functions as well as complex functionalities such as handling polyphony.

More generally, thanks to its high processing power (at least considering that it's a microcontroller) and its FPU,

the Teensy 3.6, when combined with its audio shield provides a cheap platform (>50USD) for low-latency real-time audio signal processing involving external sensors control. The lack of operating system allows for the use of low block sizes (eight samples) at a minimal computational cost, and for an extremely fast boot time (less than one second).

## 8. REFERENCES

- [1] S. Jordà, "Digital lutherie crafting musical computers for new musics' performance and improvisation," Ph.D. dissertation, Universitat Pompeu Fabra, Spain, 2005.
- [2] E. Berdahl and W. Ju, "Satellite ccrma: A musical interaction and sound synthesis platform," in *Proceedings of the New Interfaces for Musical Expression (NIME'11)*, Oslo, Norway, June 2011.
- [3] A. McPherson, "Bela: An embedded platform for low-latency feedback control of sound," *The Journal of the Acoustical Society of America*, vol. 141, no. 5, pp. 3618–3618, 2017.
- [4] E. Motuk, R. Woods, S. Bilbao, and J. McAllister, "Design methodology for real-time fpga-based sound synthesis," *IEEE Transactions on signal processing*, vol. 55, no. 12, pp. 5833–5845, 2007.
- [5] F. Pfeifle and R. Bader, "Real-time finite-difference method physical modeling of musical instruments using field-programmable gate array hardware," *Journal of the Audio Engineering Society*, vol. 63, no. 12, pp. 1001–1016, 2016.
- [6] Y. Orlarey, S. Letz, and D. Fober, *New Computational Paradigms for Computer Music*. Paris, France: Delatour, 2009, ch. "Faust: an Efficient Functional Approach to DSP Programming".
- [7] R. Michon, J. Smith, C. Chafe, S. Letz, and Y. Orlarey, "faust2api: a comprehensive api generator for android

<sup>16</sup> <https://www.raspberrypi.org/>

<sup>17</sup> <https://www.raspberrypi.org/products/raspberry-pi-3-model-a-plus/>

<sup>18</sup> <https://developer.arm.com/technologies/neon>

<sup>19</sup> 8-Channel 10-Bit ADC with SPI interface.

<sup>20</sup> <https://fe-pi.com/products/fe-pi-audio-z-v2>



- and ios,” in *Proceedings of the Linux Audio Conference (LAC-17)*, Saint-Etienne, France, May 2017, submitted for review.
- [8] V. Valimaki, J. D. Parker, L. Savioja, J. O. Smith, and J. S. Abel, “Fifty years of artificial reverberation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1421–1448, 2012.

# SOUND DESIGN THROUGH LARGE AUDIENCE INTERACTION

**Kjetil Falkenberg Hansen**  
KTH Royal Institute of Technology,  
Stockholm, Sweden

**Martin Ljungdahl-Ericsson**  
Edsbyn/University West,  
Sweden

**Ricardo Atienza**  
University College of Arts, Crafts  
and Design, Stockholm, Sweden

## ABSTRACT

In collaboration with Volvo Cars, we presented a novel design tool to a large public of approximately three million people at the three leading motor shows in 2017 in Geneva, Shanghai and New York. The purpose of the tool was to explore the relevance of interactive audio-visual strategies for supporting the development of sound environments in future silent cars, i.e., a customised sonic identity that would alter the sonic ambience for the driver and by-passers. This new tool should be able to efficiently collect non-experts' sonic preferences for different given contexts. The design process should allow for a high-level control of complex synthesised sounds. The audience interacted individually using a single-touch selection of colour from five palettes and applying it by pointing to areas in a colour-book painting showing a road scene. Each palette corresponded to a sound, and the colour nuance in the palette corresponded to certain tweaking of the sound. In effect, the user selected and altered each sound, added it to the composition, and finally would hear a mix of layered sounds based on the colouring of the scene. The installation involved large touch screens with high quality headphones. In the study presented here, we examine differences in sound preferences between two audiences and a control group, and evaluate the feasibility of the tool based on the sound designs that emerged.

## 1. INTRODUCTION

There is a growing interest in the field of sound design in the car industry; not just regarding branding and the traditional design of sounding objects (doors, motors, etc.) [1] and modern car-audio systems [2], but also the conception of outdoor/indoor sonic atmospheres linked to the emergence of silent electric cars [3]. Once the sonic trace of the combustion motor is removed, how to sonically signal the presence of the car for neighbouring pedestrians or bikers and how to conceive the new indoor sonic ambiances of the vehicle as a main qualitative component of the travel experience is a matter of regulations [4, 5], safety [6], as well as aesthetics [7].

Regarding the external sonic presence of the car, it constitutes first a key safety issue, in particular in urban en-

vironments. This sonic print can equally give support to different forms of encounter between the traversed environments and the vehicles/passengers, including informational but also masking as well as aesthetic components. A customised sound identity could operate here as a relevant mediator between car and place, and its design should thus take these multiple components into account.

This “additive” approach to everyday sonic environments (i.e., not only through traditional subtractive acoustic methods such as insulation, absorption or noise cancelling) is an expanding field of research that has been explored in the last years in different research case studies and practice-based interventions, focusing in particular on public space in dense urban contexts (e.g. [8–10]). The authors have equally explored this question of additive sound design in the context of different mobility modes e.g. in the research project ISHT (*Interior Sound Design of High Speed Trains* [11, 12]) in collaboration with the train manufacturer Bombardier. The authors have also been involved with sound design of complex shared working spaces such as flex- and activity-based offices [13]. In both examples the main focus was on how to actively improve the experience of place and situation through subtle additive sonic interventions.

Volvo Cars are focusing today on electric cars and particularly interested in the new sonic needs and possibilities brought by this more silent mobility. The world's major motor shows—the Geneva International Motor Show, New York International Auto Show, and Auto Shanghai—were regarded as a relevant opportunity for a first exploration of the users' requirements, preferences, potential desires and customisation skills. To that aim, we were asked to design the interactive environment to be used in these three locations with a visitor count of almost three million people. A general frame was provided by Volvo: *to find an intuitive and efficient way for visitors to design, or better to sketch their own sound atmosphere*; manipulating sound can certainly be regarded as an abstract and complex activity for common visitors.

As a first step towards this ambitious aim, we wanted to study here the potential efficiency of an audio-visual interactive environment for the collection of users' preferences and exploration of designing skills. However, we do not aspire to propose a finished design tool. As such, the study presented deals more with socio- or ethno-cultural differences in listening and interaction than a sound design that can readily be applied by the industry.

A simple screen-based interactive visual/graphic environment was chosen to provide support to this fast and individual prototyping process. Simplicity and robustness are

Copyright: © 2019 Kjetil Falkenberg Hansen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

central attributes in successful large-scale installations, and large touchscreens have been found to engage and captivate people [14]. Despite this simple frame, this parallel visual/sonic interaction mode represents also in a more appropriate way the multi-sensory nature of a driving experience; a purely sonic exercise would be in this sense an excessively reduced model. There are more nuanced discussions about site-specific challenges for installations [15], recommendations for design such as ease-of-use [16], fun [17], and for intuitive interaction and matching people's expectations [18].

In the last years' growing body of work on how to engage with sound and music through technology, most notably in sonic interaction design [19], sonification [20], and in new interfaces for musical expression [21], we could not find any closely related studies on designing sounds through a typical drawing task nor guidelines for such. Sketching is however a general, emerging research field within the CHI community (e.g., [22]). We are also well aware that the coupling between sound and colour or shapes have been widely investigated; in for example [23], children were drawing trajectories to fit a musical stimuli on paper, and they found support for cross-modal mapping of drawing and sound. Sonification of colours has also been found useful for visually impaired and in other applications [24], to name only one example. Our designed is also loosely inspired by a DJ system developed for creating music based on coloured discs [25]; a camera traced the spinning platter and controlled a synthesizer with the registered colours passing a tangent.

Adjacent research in synesthesia deals with cross-coupling of sensory inputs. For instance, the connection between brightness in colour and musical timbre has been shown [26], and it has been found that going from music to visual stimuli is most common [27]. Both synesthesia and related questions concerning deviation and cultural differences in colour perception [28] are somewhat relevant for this study but intentionally excluded because the method design does not really allow to take these into account.

A soundscape is according to [29] the auditory counterpart to a visual landscape. Soundscapes can exist as perceptual constructs, but also as physical phenomena [30]. When perceived in a shared context, the constituents of the acoustic and the visual interrelate [31].

By essentially using a number of shared composition materials, we want to explore how different set contexts influence the choices realised by participants. Is it possible to trace case-specific preferences? Along this process, the user could only manipulate sound by interacting with a colour book interface. All interaction with the system and the resulting compositions were logged in the form of text files and recordings collecting the series of actions executed by visitors on the tactile screen.

## 2. METHOD

The interactive system we built, called *Volvo Sound Studio*, was based on an audio-visual environment including touch screens, high-end headphones, and software built entirely in Pure data and the graphic library GEM (for details

Hardware/ software	Description	Link
Windows	Surface Studio PixelSense™ 28" touch display, Windows 10	<a href="#">↗</a>
Bowers & Wilkins	P9 Signature over-ear headphones	<a href="#">↗</a>
Audio-Quest	DragonFly Red 32 bit ESS 9016 DAC and headphone amplifier	<a href="#">↗</a>
Pure Data 0.47-1	Real-time programming environment for audio	<a href="#">↗</a>
GEM 0.93.3	Graphics environment for multimedia	<a href="#">↗</a>

Table 1. Overview of the technical set-up and system components.

see Table 1). While the hardware was decided by or in collaboration with Volvo and Bower & Wilkins (Volvo's partners in the design of their car-audio systems and main providers of audio components) the programming environment and the interaction design were the responsibility of the authors. The system has previously been described in brief [32], but then without results from the interaction.

### 2.1 Interface and Interaction Design

During several brainstorming and development sessions together with representatives from the partner companies, the concept of having a simplistic colour-book interface was decided. A strong argument was to create an experience that did not resemble sound design or mixing tasks, and something that would appear novel, which is of utmost importance for a large-scale public event like this. Another key characteristic of this concept was the accessible and intuitive nature of the interaction to happen, requiring no previous experience in editing or designing sound, no specific introduction to the user and very limited time before reaching a meaningful result. The main focus was thus placed on the driving contexts explored (graphically presented on screen and sonically evoked through a soundscape background track) and their specific impact on the sound choices operated by visitors.

The concept of the colour-book, with its deep abstraction of a represented environment and its simple interactive mode, was intended to allow the user not to focus exclusively on the visual information, but also on the sound textures under development, i.e., in search of a balance between both senses involved. A highly demanding visual experience could more easily become a mono-sensorial exercise from an attention point of view. In the same sense, a broader palette of graphic possibilities and realism would equally require a careful analysis of the correlations proposed between colour and sound; this complex aspect was avoided as this early study focused on designing potentials more than final sketching tools.

Participants were invited to "paint" three different land-

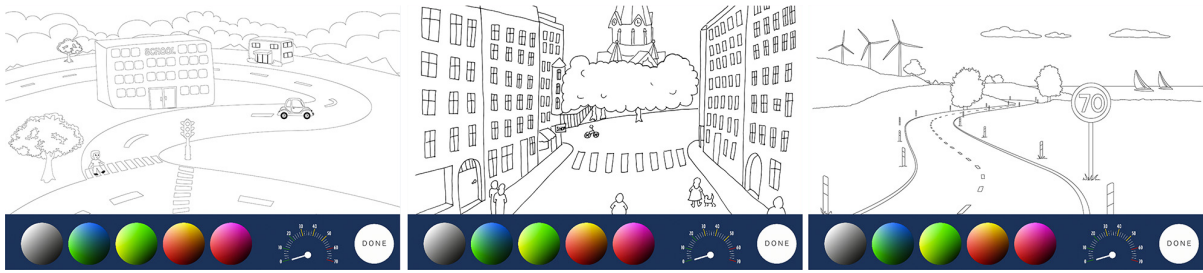


Figure 1. The three scenes presented to the audience in colour-book style: from left the school area, city centre and countryside road scenes. Colour is chosen in the palettes and then applied to an empty area. Each palette has a sound assigned, and the sounds from the painted areas are mixed. The speedometer changes a global filter on the mixed sound.

scapes represented by three stylised scenes (see Fig. 1), namely a *school area scene* with a school building surrounded by curvy small roads and a low density area, a *city centre scene* with a dense urban environment, and a *countryside scene* depicting a curvy road in an open semi-natural landscape. None of the scenes included presence of other cars, neither traffic or congestion, as we wanted users' attention to be only on their "own" car. In Fig. 1, the car in the school scene can however indeed be interpreted to be another sounding car.

The graphic language employed is intentionally simple, based on the model of the colour-book, as well as the colouring mode in action: each tone is applied at once on entire predefined areas of each drawing, clearly delimited by black lines on a white canvas. In the version shown to the public, the user could choose from five colour palettes (gradations of grey, green-blue, green-yellow, red-yellow and red-magenta) and apply the chosen tone to seven different areas in the drawing. The selection of the colour tone was providing via headphones an immediate sonic feedback varying as the user was exploring the different possible colour nuances; different filters and sound effects followed, in sonic terms, the movement of the arrow on the colour selection area (see Fig. 2).

Each colour palette was linked to a different sound material, the same per palette for the three scenes explored (see Table 2). While selecting the colour, i.e., working on a colour palette, only the corresponding sound can be heard, and when finally applied to the main canvas, the user gets the entire sonic mix or composition, including a background atmosphere specifically designed for each individual scene configured by a number of typical sonic components of the depicted landscape (activities, textures, soundmarks, etc.). This background atmosphere or soundscape aims at providing a coherent and easily understandable sound supporting layer for a more flexible and free exploration of the new materials to insert which essentially present no specific relation to the different contexts. The exception is three specific contextual sounds presenting, *a priori*, an intentional relation to each context; these three sounds will be the object of particular attention along the analysis. At any time, the visitor can interact with a speedometer, described below.

When no interaction was provided, the screen automatically displayed an information video silently demonstrat-

ing how to use it,<sup>1</sup> and inviting passers-by to test the environment. For sound samples of both countryside scenes represented in Fig. 2, including the specific background soundscape (final compositions, produced by participants), see Table 3.

## 2.2 Sound Design and Synthesis

A total of five sounds were designed. Each was five seconds long and could be seamlessly looped at any given point. The composing of the sound material had to cater two design specifications. Each sound should easily and harmoniously have the ability to be intertwined with any other of the sounds. The sounds also needed to possess the right balance between being interesting without generating irritation during prolonged listening. Three of the sounds would also be contextually linked to the scene through similarity to the matching soundscape, one for each scene, and two would be contextually deviant. The divergent sounds were composed with the intention to be associated with a real and a fictional vehicle: one sound was an actual Volvo engine and the other a paraphrasing of how UFOs often are depicted in sci-fi movies. This was done to find out if the participants would connect the contextually connected sounds to the scene and if an actual car engine were preferred.

The sounds, except the engine sound, were created with an energy concentration around 1000 Hz, a frequency range not too crowded in the exhibition areas. The sounds were also designed through rhythm and pitch changes, making them fluctuate in order to be distinguished from other constituents emitting in the same frequency range of the soundscape. The intention of the dynamic character of the sounds was to be more cohesive with the dynamics of the ever-changing soundscape. The sounds would fit the surrounding rhythmic events and at the same time convey movement and grab attention, and avoid being static sounds that add to the denseness of the urban sonic environment.

The sound design had the aim of letting the user design its preferred car sound based on the location generating the driving situation, in this case the sonic and visual stimuli presented at the scene. E.g., the harmonic sound was designed to be connected to the city center scene, having a dynamic movement and a melodic nature to be discerned

<sup>1</sup> Video: <https://kth.box.com/v/smc2019-HLA-video>



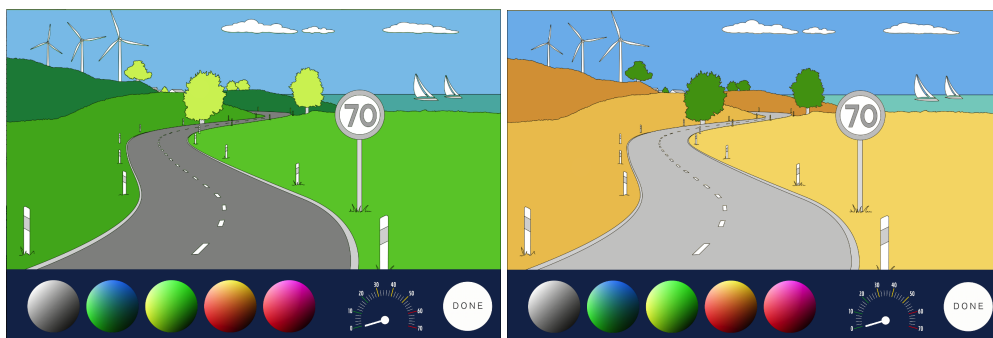


Figure 2. Examples of painted countryside scenes with different interpretation of the vegetation. The left image would be dominated by sounds from the green palette, namely the “rolling” sound, while the right image would be dominated by sounds from the orange palette, namely the “motor” sound.

in that context. The texture of a church bell used was softened to get a floating and non intrusive sound, that for the listener would alter between staying in the background and foreground.

The sounds described above were not played as is, but instead treated in a synthesis based on looping. Each instance of choosing colour tone in a palette and applying to an empty space would trigger the synthesizer. In order for the result to appear dynamic (like a motor sound), the sound was replayed with three very short loops (fractions of a second) of random start and length within the sound file from its start, middle and end. Onto each looped part, we added an audio effect controlled by the colour nuance (see Table 2). In total there could thus be 21 loops going simultaneously, creating a very dense but still dynamic composition.

In order to provide a coherent visual/aural experience, once a colour is applied onto a scene, it is placed in the stereophonic space according to its relative position in the drawing. Each area in a scene will thus correspond to a different left/right panning value. This effect will be particularly audible in the binaural space supported by headphones.

The speedometer aims at evoking the familiar glissando sound effect linked to variations of speed as the motor accelerates or decelerates. To that extent, the speedometer controls a non-linear pitch-shifting function and loop-speed effect where pitch and loop speed do not covary. Each context will present a speed limit, according to normal traffic regulations (urban, country road, etc.).

### 2.3 Data Collection

Data were collected from two of the three expositions we exhibited the Sound Studio in; Geneva and Shanghai, as well as from a supplementary laboratory experiment.<sup>2</sup> Due to reasons beyond our control, New York data became inaccessible for this study. We chose two dates for inclusion: both Sundays after the first open weekend (March 3 and April 4 2017, respectively). These days were less busy and we expected the installation to be stable (it turned out they ran without trouble for the entire duration).

<sup>2</sup> Log data: <https://kth.box.com/v/smc2019-HLA-data>

There are many parameters from such data collection that we have little or no control over, and several are important. The following list is inconclusive, but can serve as a careful suggestion for reading the results with some attention:

**The user:** We are confronted in this study to entirely anonymous participants: no personal information regarding the user and its profile was registered. That includes nationality, gender, age, physical characteristics, background, etc.

**The intention:** The users were not debriefed in order to understand their interaction. This means we know nothing about if there was preference for using colour instead of sound, the users seriousness, if they were happy with the results, or if they changed all parameters from their desired soundscape in the last moment before submitting.<sup>3</sup>

**The interaction:** Nothing surrounding the session apart from the time of day was registered. That includes if it was a returning user, if headphones were used correctly or even used at all, if only one user interacted, if there were disturbances elsewhere, etc.

However, there were always exhibition hosts attending the two screens and giving support to the guests, so we can reasonably argue in favour the exploitable nature of the data collected. In order to verify this hypothesis, we conducted—in addition to the exhibition setting—a laboratory experiment with twelve first year media technology students. They interacted without disturbance for as long as they wished. The group was balanced with regards to gender (6f/6m), but unbalanced with respect to age (between 19–25 years), nationality (Swedish), and all being technology students. The purpose of having a control group was to understand whether stress and contextual factors (time, sound environment, multiple surrounding presences and actions) would critically influence the exhibition audiences.

### 3. RESULTS

For the two chosen dates from the data collection period, we have 312 and 431 sessions from Geneva and Shanghai, respectively, or one session every two minutes. A session was defined as one completed soundscape design from selecting a scenery to clicking the ‘Done’ button. The av-

<sup>3</sup> Such behaviour would however be identified in the log data.

Sound	Description and effect	Context	Palette	Link
“Harmonic”	Resonating and reverberating church bell texture fragments and a minor chord. Modified with pitch-shifted delay.	City centre	Grey–black	<a href="#">↗</a>
“Rolling”	Filtering and delaying a minor 7 <sup>th</sup> chord using a recording of waves to trigger the delay channel for an organic rhythm. Modified with flanger.	Countryside	Green–blue	<a href="#">↗</a>
“School bell”	A recording of wind chimes modulated to tremolate. Modified with a time-varied delay line [33].	School area	Green–yellow	<a href="#">↗</a>
“Motor”	A recording of a Volvo V60 internal combustion engine high-pass filtered and moderately distorted to accentuate the engine sound. Modified with a comb-filter octaver.		Red–yellow	<a href="#">↗</a>
“Sci-fi”	Oscillating a single tone which then is reverberated, and further flanged and resonated. Modified with voltage-controlled bandpass filter sweeps.		Red–magenta	<a href="#">↗</a>
School soundscape	A mix of field recordings from a playground, a quiet urban environment, and a parking lot outside a shopping mall.			<a href="#">↗</a>
City soundscape	A mix of field recordings of two cities recorded at different locations early in the morning, a recording from a busy pedestrian street, and a church bell.			<a href="#">↗</a>
Countryside soundscape	A mix of field recordings of breaking waves, seagulls, cars driving at a distance, and soft wind on a field.			<a href="#">↗</a>

Table 2. Sounds used in the interface with arbitrary names. The effect was added to the sound corresponding to the colour nuance. The links go to sound examples in the Soundcloud repository.

erage time for the interacting visitor was 81 seconds, so with 2–4 computers in the exhibition space, they were almost constantly in use. The longest session lasted 430 seconds, and all sessions shorter than 15 seconds were removed from analysis before the estimation made above. The control group generated 35 sessions of 36 planned.

There were significant differences ( $p \ll 0.001$ , two-tail t-test) in average time spent on interaction, where the Shanghai audience spend 19% more time, or around 14 seconds longer per completed session. Also, this audience painted in average two more areas in a session than the Geneva audience (the areas were painted 17.3 and 14.9 times, respectively,  $p = 0.017$ ). In average, the Shanghai and Geneva groups adjusted and applied one new sound every five seconds. The control group had longer interaction time (44% longer,  $p = 0.006$ ), but did not paint more areas; in effect, they listened more than two seconds longer to each sound before applying it.

The most prominent difference in interaction behaviour is connected to exploring the sound palette. While we cannot directly measure how the users listen for changes in sound, the logging function produces one line for each incremental value from dragging the finger within the palette. The exhibition audiences have similar values—Geneva almost twice as many as Shanghai—while the line count of the control group is more than six times higher than that.

In each scene, three of the five sounds (corresponding to the five colour palettes described above and shown in Table 2) were used similarly by the Geneva and Shanghai audience; the harmonic, the rolling, and the sci-fi sounds (see Fig. 3). The school bell sound was favoured only by the

Geneva audience, and only in the School scene (by 89%,  $p \ll 0.001$ ). We found no difference between Shanghai and the control group. The motor sound was favoured by the Shanghai audience, but only in the City scene (by 59%,  $p \ll 0.001$ ). This preference was even stronger in the case of the control group (82% more,  $p \ll 0.001$ ).

The three contextual sounds (the school bell, the harmonic, and the rolling sound) each had an intended connection to the scenes (the school, the countryside, and the city, respectively). For the school bell sound/school scene, the Geneva audience were, as shown above, significantly more likely to choose the contextual sound (almost twice as much). For the other scenes, there were no differences.

The audience in Geneva seemed to consistently choose a higher speed on the speedometer than those in Shanghai (in average 13.4% and 8.9% faster than “neutral” speed, respectively). However, the significance test did not confirm this observed tendency.

By shifting the colour nuance on the palette, each sound could be ‘tweaked’ in the sense that the sound was manipulated with an increasingly noticeable sound effect. Colour nuance was mapped to position in the circle, which showed a colour grading, while distance from the centre of the palette corresponded to sound effect strength regardless of absolute position in the circle. There was no significant difference between the audiences nor the control group in sound effect strength. In average, they tweaked the sounds to 0.56 distance from the centre, or just above halfway out towards the edge.

We did not find any difference in how varied the sound designs were in terms of how many palettes that were used

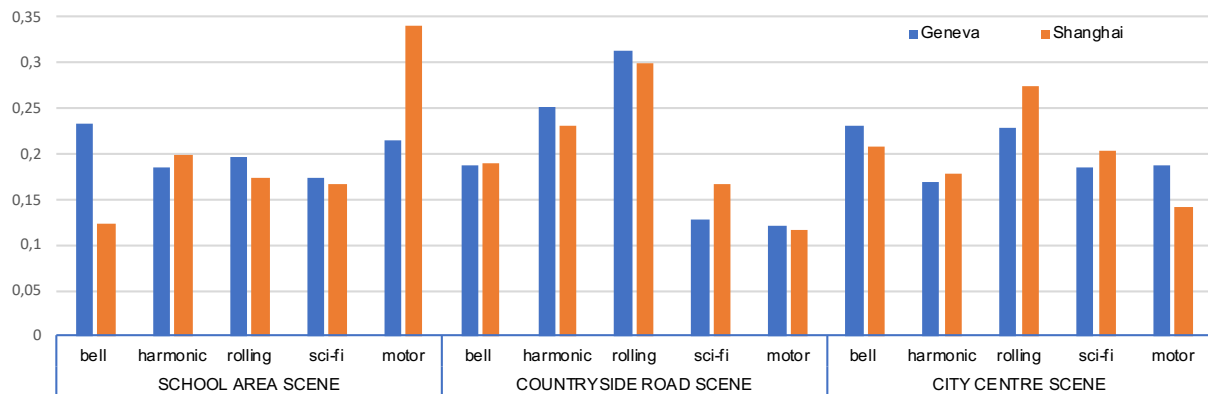


Figure 3. Proportion of sounds used by Geneva and Shanghai visitors for each of the three scenes.

for each session. In average, 2.9 of the five palettes were used for one painting. However, the control group showed a larger variation with 3.5 palettes used in average ( $p = 0.005$ ). Even for this measurement there is ambiguity concerning whether the user focused on sound or colour.

#### 4. DISCUSSION

From the mentioned uncertainties concerning data collection, we cannot *know* if the users of our system among the audiences at either Geneva or Shanghai are representative for those geographically distant places. However, judging from the partner companies present, exhibition hosts, and popular media reports, we find it is reasonable to assume that visitors are representative for the general population of those places. In the following, we generalise the results accordingly. Furthermore, the statistical analysis must be read with caution as the experiment lacked in control. Thus, only a few distinct findings were reported above and included here to serve as a basis of discussion and future work.

Among the measurements extracted from the log files, there are far more similarities than differences between Geneva and Shanghai. They adjust the speed similarly, they tweak the sound to the same extent, and they use a comparable varied selection of palettes for a given scene. They do however differ in the amount of effort and time that is put into the interaction.

Five designed “base sounds” were assigned to the palettes, of which three were specifically suited for each of the three scenes. For some reason, the school bell sound was strongly associated with the school scene by the Geneva audience, but not the Shanghai. The other sound that the audiences applied differently was the motor sound. Generally, the Shanghai audience chose this for the school scene when the Geneva chose the bell sound.

One possible explanation is that the sound design was done by a European who has more experience with school bell sounds in Europe than in Asia. Another explanation is that a bell sound is not common in traffic situations in Shanghai, while low-frequency motor sounds are. Furthermore, although unlikely, we cannot overlook the possibility that users only chose a colour to match the scene which

is different between Geneva and Shanghai.

The control group was in many respects more similar to the Shanghai audience, which contradicts the above-mentioned regional differences in how one experiences the school bell and motor sounds. Also, the control group favoured the rolling sound. An alternative explanation to these found differences could be that in the Geneva exhibition there were surrounding sounds which interfered with the low-frequency motor and rolling sounds.

A further perspective on choice of sounds is the sound feedback from the act of choosing a colour. For instance, to paint the sky blue, the user needs to find the colour in the second palette, or the “rolling” sound. Furthermore, to get to the blue, the audio effect will be strong (for sound examples demonstrating this, see Table 3). As a result, the user needs to compromise between image naturalness or intention and how it sounds. Similarly, the easiest way to paint a grey or black road is to use the first palette with the “harmonic” sound. However, none of these speculations can really explain why the Geneva public favoured the school bell sound (greens) for the school area.

Overall, the Shanghai audience spent more time and repainted more areas, despite that the number of completed sessions was higher. Without having time constraints or other distractions, the control group devoted considerably longer for finishing a session than the exhibition audiences, but without painting more areas. Also, the control group listened or searched far more attentively for the right nuance in the sound/colour palette. Regardless of this, it seems that most users could finish a sound design within less than 90 seconds. Considering the substantial variation in sound that was possible with the relatively simple tool, we are intrigued by this finding.

#### 5. CONCLUSIONS

The installation *Volvo Sound Studio* globally fulfilled the expectations in terms of reliability as well as efficiency of planned interaction. Visitors to the major car exhibitions in Geneva and Shanghai (as well as New York, not covered in this paper) could complete rather complex sound designs within a period of time as brief as a minute. It was efficient also in regard to the proved intuitive nature

of the interaction; granting universal access to visitors regardless of their skills or age. The graphic codes chosen and disposition of elements on the screen naturally guided the users in their explorations according to the observations realised. The stability of the system proved to be entirely reliable, which is not obvious when it comes to interactive systems running non-stop along several days and with no particularly qualified local maintenance.

We found that audiences at the Geneva and Shanghai motor shows had different preferences for the car's soundscape outside a school and in a city centre. The Geneva users favoured a school-bell sound which was specifically designed to match that scene, and for the city scene, the Shanghai users favoured a sound based on a combustion engine recording.

In future studies we would like to explore a number of different choices regarding aspects such as alternative programming environments (GEM could readily be replaced by better performing graphic options such as Processing), data collected on participants (details on gender, age, would have been relevant criteria to explore, even perhaps the possibility of short surveys as initially suggested to our industrial partner) or even complementary methods focusing only the aural dimension to be compared with the results obtained here (e.g. employing the same sound materials during the design process).

With regards to carrying out research within the confines of a prominent publicity effort made at the most important venue for one of the leading car producers, there are naturally some compromises that had to be made. As a first general reflection, the design of the interactive modes and data collection protocols would have certainly been conceived differently within a purely research frame; efficiency was here a key concept, with a suggested time limit of one minute per user. Without doubt, such compromises are typically in conflict with research needs and requirements.

On a similar key, the installation we ended up building is probably not the ultimate for finding answers to how people design sounds: the colour-book concept could have been exchanged for other concepts. The design process was limited to the effort of the authors and with influence from a few involved persons at Volvo. Also, it would be impractical or even impossible to gather personal data from the visitors. On the other hand, there are very few opportunities of reaching out to such a number of people in another context.

### Links to Sound Examples

All sound examples have been uploaded to the Soundcloud repository: <https://soundcloud.com/kjetil-falkenberg-hansen/sets/sound-examples-smc2019>; see Table 3.

### Acknowledgments

The authors would like to thank Annika Hedin and Fredrik Hagman at Volvo Cars and Marie Hernmarck, Susanne Park and Robert Tenggren at SCP Grey for the productive conversations and helpful input. The project was sponsored by Volvo Cars.

Description	Linkname
The harmonic sound	harmonic-sound
The rolling sound	rolling-sound
The school bell sound	schoolbell-sound
The motor sound	motor-sound
The sci-fi sound	scifi-sound
School area soundscape	soundscape-school
City centre soundscape	soundscape-city
Countryside soundscape	soundscape-country
Finished scene Figure 2a	scene2a
Finished scene Figure 2b	scene2b
Speedometer demonstration from slow to fast to slow	speedometer
Audio effect demonstration on five sounds from minimum to maximum	audio-effects

Table 3. List of sound examples uploaded to the Soundcloud repository. All links have the format <https://soundcloud.com/kjetil-falkenberg-hansen/linkname>

## 6. REFERENCES

- [1] E. Cleophas and K. Bijsterveld, *Selling Sound: Testing, Designing, and Marketing Sound in the European Car Industry*. Oxford University Press, dec 2011.
- [2] R. Coppola and M. Morisio, "Connected car," *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–36, oct 2016.
- [3] D. S. Kim, R. W. Emerson, K. Naghshineh, and K. Myers, "Influence of ambient sound fluctuations on the crossing decisions of pedestrians who are visually impaired: implications for setting a minimum sound level for quiet vehicles," *Journal of Visual Impairment & Blindness (Online)*, vol. 108, no. 5, p. 368, 2014.
- [4] National Highway Traffic Safety Administration (NHTSA), *Federal Motor Vehicle Safety Standards; Minimum Sound Requirements for Hybrid and Electric Vehicles*. US Government Federal Register, 2016, document Citation: 81 FR 90416. [Online]. Available: <https://www.federalregister.gov/d/2016-28804>
- [5] Inland Transport Committee, *Proposal for a new Regulation concerning the approval of quiet road transport vehicles (QRTV)*. United Nations, 2016, eCE/TRANS/WP.29/2016/26. [Online]. Available: <https://www.unece.org/fileadmin/DAM/trans/doc/2016/wp29/ECE-TRANS-WP29-2016-026e.pdf>
- [6] A. Zeitler, "Psychoacoustic requirements for warning sounds of quiet cars," *SAE International Journal of Passenger Cars - Electronic and Electrical Systems*, vol. 5, no. 2, pp. 572–578, jun 2012.
- [7] N. Misdariis, A. Cera, E. Levallois, and C. Locqueteau, "Do electric cars have to make noise? An emblematic opportunity for designing sounds and soundscapes," in *Acoustics 2012*, S. F. d'Acoustique, Ed., Nantes, France, Apr. 2012. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-00810920>



- [8] B. Hellström, "Noise design : Architectural modelling and the aesthetics of urban acoustic space," Ph.D. dissertation, KTH, School of Architecture, 2003, nR 20140805.
- [9] M. E. Nilsson, J. Alvarsson, M. Rådsten-Ekman, and K. Bolin, "Auditory masking of wanted and unwanted sounds in a city park," *Noise Control Engineering Journal*, vol. 58, no. 5, p. 524, 2010.
- [10] B. Hellström, M. E. Nilsson, Ö. Axelsson, and P. Lundén, "Acoustic design artifacts and methods for urban soundscapes : A case study on the qualitative dimensions of sounds," *Journal of Architectural and Planning Research*, vol. 31, no. 1, pp. 57–71, 2014, qC 20140625.
- [11] R. Atienza and N. Billström, "Fighting "noise"=adding "noise"? – Active improvement of high-speed train sonic ambiances," in *Ambiances in action/Ambiances en acte (s)-International Congress on Ambiances, Montreal 2012*. International Ambiances Network, 2012, pp. 235–240.
- [12] K. F. Hansen and R. Bresin, "Use of soundscapes for providing information about distance left in train journeys," in *Proceedings of the 9th Sound and Music Computing Conference*, Copenhagen, jul 2012, pp. 79–84. [Online]. Available: <http://www.speech.kth.se/prod/publications/files/3764.pdf>
- [13] M. L. Eriksson, R. Atienza, and L. Pareto, "The Sound Bubble: A context-sensitive space in the space," *Organised Sound*, vol. 22, no. 1, pp. 130–139, mar 2017.
- [14] C. Ardito, P. Buono, M. F. Costabile, and G. Desolda, "Interaction with large displays," *ACM Computing Surveys*, vol. 47, no. 3, pp. 1–38, feb 2015.
- [15] K. Beilharz and A. Martin, "The 'interface' in site-specific sound installation," G. Essl, B. Gillespie, M. Gurevich, and S. O'Modhrain, Eds. Ann Arbor, Michigan: University of Michigan, May 21-23 2012.
- [16] J. Hochenbaum, O. Vallis, D. Diakopoulos, J. Murphy, and A. Kapur, "Designing expressive musical interfaces for tabletop surfaces," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Sydney, Australia, 2010, pp. 315–318. [Online]. Available: [http://www.nime.org/proceedings/2010/nime2010\\_315.pdf](http://www.nime.org/proceedings/2010/nime2010_315.pdf)
- [17] B. Shneiderman, "Designing for fun," *interactions*, vol. 11, no. 5, p. 48, sep 2004.
- [18] L. Hespanhol and M. Tomitsch, "Strategies for intuitive interaction in public urban spaces," *Interacting with Computers*, vol. 27, no. 3, pp. 311–326, jan 2015.
- [19] K. Franinović and S. Serafin, Eds., *Sonic Interaction Design*. MIT Press Ltd, 2013.
- [20] S. Pauletto, H. Cambridge, and P. Susini, "Data sonification and sound design in interactive systems," *International Journal of Human-Computer Studies*, vol. 85, pp. 1–3, jan 2016.
- [21] R. Medeiros, F. Calegario, G. Cabral, and G. Ramalho, "Challenges in designing new interfaces for musical expression," in *Design, User Experience, and Usability. Theories, Methods, and Tools for Designing the User Experience*. Springer International Publishing, 2014, pp. 643–652.
- [22] M. Sturdee, M. Lewis, and N. Marquardt, "Feeling SketCHI?" *Interactions*, vol. 25, no. 6, pp. 64–69, oct 2018.
- [23] E. Frid, R. Bresin, P. Alborn, and L. Elblaus, "Interactive sonification of spontaneous movement of children – cross-modal mapping and the perception of body movement qualities through sound," *Frontiers in Neuroscience*, vol. 10, nov 2016.
- [24] S. Cavaco, J. T. Henriques, M. Mengucci, N. Correia, and F. Medeiros, "Color sonification for the visually impaired," *Procedia Technology*, vol. 9, pp. 1048–1057, 2013.
- [25] N. Villar, H. Gellersen, M. Jervis, and A. Lang, "The ColorDex DJ system: a new interface for live music mixing," in *Proc. of the Conference on New interfaces for Musical Expression*. New York: ACM, 2007, pp. 264–269.
- [26] L. E. Marks, "Synesthesia and the arts," in *Cognitive Processes in the Perception of Art*. Elsevier, 1984, pp. 427–447.
- [27] G. Berman, "Synesthesia and the arts," *Leonardo*, vol. 32, no. 1, pp. 15–22, feb 1999.
- [28] R. Pettersson, "Cultural differences in the perception of image and color in pictures," *ECTJ*, vol. 30, no. 1, pp. 43–53, 1982.
- [29] M. Raimbault and D. Dubois, "Urban soundscapes: Experiences and knowledge," *Cities*, vol. 22, no. 5, pp. 339–350, oct 2005.
- [30] J. Appleton, *The experience of landscape*. New York: J. Wiley & Sons, 1996.
- [31] J. L. Carles, I. L. Barrio, and J. V. de Lucio, "Sound influence on landscape values," *Landscape and Urban Planning*, vol. 43, no. 4, pp. 191–200, jan 1999.
- [32] K. F. Hansen, M. L. Eriksson, and R. Atienza, "Large-scale interaction with a sound installation as a design tool," in *Proceedings of the 12th International Audio Mostly Conference on Augmented and Participatory Sound and Music Experiences - AM '17*. ACM Press, 2017.
- [33] P. Massat, "GuitarExtended: DownTheDrain," Pure data effects patch, 2012.

# EVALUATING A CONTINUOUS SONIC INTERACTION: COMPARING A PERFORMABLE ACOUSTIC AND DIGITAL EVERYDAY SOUND

**Fiona Keenan**

Department of Theatre, Film and  
Television  
University of York, UK  
fiona.keenan@york.ac.uk

**Sandra Pauletto**

Department of Media Technology  
and Interaction Design  
KTH Royal Institute of Technology, Sweden  
pauletto@kth.se

## ABSTRACT

This paper reports on the procedure and results of an experiment to evaluate a continuous sonic interaction with an everyday wind-like sound created by both acoustic and digital means. The interaction is facilitated by a mechanical theatre sound effect, an acoustic wind machine, which is performed by participants. This work is part of wider research into the potential of theatre sound effect designs as a means to study multisensory feedback and continuous sonic interactions. An acoustic wind machine is a mechanical device that affords a simple rotational gesture to a performer; turning its crank handle at varying speeds produces a wind-like sound. A prototype digital model of a working acoustic wind machine is programmed, and the acoustic interface drives the digital model in performance, preserving the same tactile and kinaesthetic feedback across the continuous sonic interactions. Participants' performances are elicited with sound stimuli produced from simple gestural performances of the wind-like sounds. The results of this study show that the acoustic wind machine is rated as significantly easier to play than its digital counterpart. Acoustical analysis of the corpus of participants' performances suggests that the mechanism of the wind machine interface may play a role in guiding their rotational gestures.

## 1. BACKGROUND

This evaluation was conducted as part of an investigation into the sonic interactivity of historical theatre sound effects, devices created for soundmaking through performance actions, mechanisms and materials in the late nineteenth and early twentieth century. It is proposed that as interactive mechanisms designed explicitly for the performance of everyday sound events such as rain, wind and thunder, theatre sound effects offer the opportunity to explore how very simple hand actions might be coupled to the performance of complex digital sounds in a perceptually meaningful way. An examination of historical sources on theatre sound effects has shown that these historical interfaces were created using an

approach much like Franinović's proposed *enactive sound design* [1]. This is a Sonic Interaction Design (SID) strategy that engages with the potential of ergoaudition (listening to self-produced sound) [2] to facilitate learning in a sonic interaction. Sound is produced directly and continuously through a user's movement, guides their sensorimotor activity and allows them to build on previously accumulated tacit knowledge of action and sound [3,4]. With no established system of sound notation in use in theatres in the late nineteenth and early twentieth century, sound effects were explicitly designed to facilitate the development of bodily skill in sound performance through a simple process of exploration and rehearsal while listening to self-produced sound.

As simple acoustic interfaces that produce the *effect* of a familiar everyday sound [5] in performance, theatre sound effect designs also afford an exploration of the perceptual experience of a continuous sonic interaction, and potentially expressive sound performance, without the need for participants to have a particular level of prior musical experience. This research therefore adapts evaluation methods from previous research into the design of Digital Musical Instruments (DMIs) focused on musical expression [6–8], and applies them to a broader cohort of participants. The evaluation method presented here also positions theatre sound effect designs as a potentially useful means of controlling and comparing specific modes of multisensory feedback in the evaluation of a continuous sonic interaction [9, 10]. To examine how the enactive qualities of specific historical theatre sound effects might be uncovered and then captured in the design of a continuous sonic interaction with a digital sound, this research focused on exploring the experience of a continuous sonic interaction with one acoustic theatre sound effect, and comparing this experience with that afforded by a digital model of its sonic feedback in performance. This work extends the methodology used in prior research in the field of SID, which examined the enactive qualities of Luigi Russolo's *intonarumori* family of early twentieth century acoustic noise instruments in order to recreate them as digital models [11].

### 1.1 Interface Design and Synthesis Method

This work began with the construction of a working example of a theatre sound effect, an acoustic theatre wind machine, from historical design instructions. A

Copyright: © 2019 Fiona Keenan et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

wind machine consists of a wooden slatted cylinder, which is mounted on a central axle and A-frame and covered by a cloth. A crank handle coupled to the axle allows a performer to rotate the cylinder. As the handle is turned, the wooden slats of the cylinder rub and scrape the encompassing cloth, which produces a wind-like sound. (Figure 1). This acoustically modelled everyday sound [5] can have perceivably repetitive and machine-like qualities at slow and regular speeds of rotation, but when activated with a gesture of continuously varying speed the sound becomes more convincing as a wind effect. The cylinder of the wind machine has flywheel qualities, storing rotational energy and resisting changes in rotational speed during a performance with its crank handle. This adds a complex sensation of shifting weight and effort to the very simple rotational gesture.

Following an exploration of its process of sound production, a prototype digital model of this working wind machine was programmed in Max/MSP<sup>1</sup>. This digital prototype was created using a procedural approach to sound modelling [12]. Rather than designing a performable wind-like sound from a physical model of real-world aeroacoustics [13, 14], or a signal-based method using noise and a band-pass filter [12], this prototype aimed to directly model the mechanical wind effect, i.e. the theatre wind machine's wooden slats and cloth that interact to produce sound. For this reason, the model was based on the rubbing and scraping interaction between each wooden slat and the encompassing cloth of the wind machine during a rotational gesture performed with the crank handle (Figure 2). In this way, the mechanical design of the acoustic wind machine and the physics inherent in its sound production could be explored through the modelling process, a method long in use in musical acoustics [15]. The perceptual experience and potential distinctions between real-world wind sounds and the cloth-based effect of the acoustic wind machine could also be examined, and the primacy of the performer's gesture in the realism of the wind effect could be transferred more explicitly to the digital prototype.

Twelve instances of the Sound Design Toolkit (SDT) physical model of friction [16] were implemented in Max/MSP to represent each of the twelve slats of the acoustic wind machine and their interaction with the cloth. Some additional dispersion of the resulting friction sound through each side of the cloth was also implemented using a digital waveguide [17]. The acoustic wind machine's mechanism was fitted with a rotary encoder, some laser-cut gearing and an Arduino<sup>2</sup>, to capture data from its rotational motion. This allowed the acoustic wind machine's crank handle to drive the digital model of its sound in performance. The rotary encoder's data was mapped to each of the twelve digital slat models, which were activated according to the position of the wooden slats on the acoustic wind machine. The rotational data also slightly modulated the delay time to the cloth model to add some of the characteristic whistling of the

acoustic wind machine at high rotational speeds to its digital counterpart.

Using the acoustic wind machine as a performance interface for the digital model in Max/MSP maintained a consistent tactile and kinaesthetic feedback during a performance of both the acoustic and digital wind-like sounds. It also allowed the acoustic and digital wind-like sounds to be simultaneously activated by the same performance gesture, facilitating an acoustic analysis and comparison of the acoustic and digital sounds that helped develop and calibrate the digital model in Max/MSP. This objective analysis confirmed that the digital model was quite similar to the acoustic wind machine, particularly at slow and regular speeds of rotation. The stages of this work, and the full technical details of the digital model, have been previously described elsewhere [18, 19].



Figure 1. The working acoustic wind machine.

## 2. EXPERIMENT DESIGN

With the acoustic wind machine producing its own wind-like sound and simultaneously driving its digital counterpart during performance, it was possible to design an experimental procedure to evaluate only one modality of the interaction - the sonic feedback itself. The evaluation was focused on exploring whether the continuous sonic interaction with the digital wind-like sound was perceivably 'similar enough' to that of its acoustic counterpart. If so, this would confirm that the digital model had captured many of the sonic qualities of the acoustic wind machine, and could be used as a substitute in a future evaluation. If not, the evaluation would help to determine how the digital model of the wind-like sound should be developed further. Comparing the two sonic interactions would also help to discover more about the perceptual experience of performing an everyday sound [5], and establish a baseline of results against which future evaluations could be compared.

In the absence of prior work specifically examining the sonic interactivity of theatre sound effects in performance, this evaluation aimed to establish statistically significant

<sup>1</sup> <http://cycling74.com/>

<sup>2</sup> <https://www.arduino.cc/>

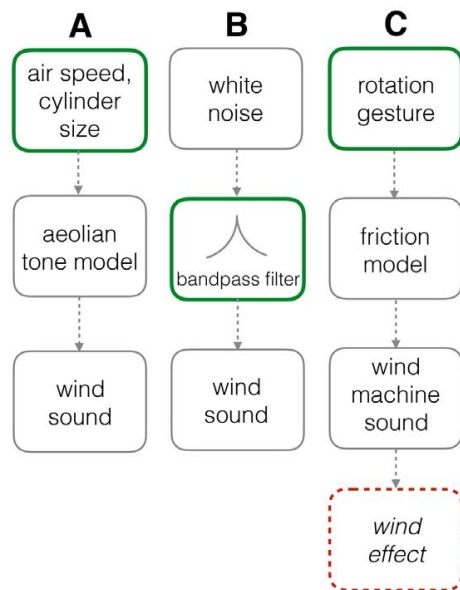


Figure 2. A comparison of approaches to digitally synthesizing a wind sound using A) a physical model of real-world wind [13, 14], B) a signal-based approach [12], and C) this research. The green outline denotes performance data mapping.

results while also collecting some qualitative data in the form of participants' free descriptions of their experiences of performing the acoustic and digital wind-like sounds. To clearly investigate whether participants might perceive a particular rotational gesture of the crank handle in the acoustic or digital wind-like sound, the experimental design focused on *operationalizing* the experience of a continuous sonic interaction with both the acoustic and digital wind-like sounds. This follows prior research in the field of Digital Musical Instrument (DMI) Design, where musical performers were given defined audio cues to imitate, and time to reflect on their performance experiences, when evaluating a new DMI [6]. This would help to examine whether participants could understand a rotational gesture from the wind-like sounds they heard, and then translate this to a performance gesture of their own.

Previous work to acoustically evaluate and compare both wind-like sounds found that the prototype digital model's response was closer to that of its acoustic counterpart at slower and more regular speeds [19]. As such, this evaluation focused on simple and steady performance gestures. The sounds produced by these gestures were used as stimuli to elicit participants' performances. Participants were also asked to reflect on how they felt their performances compared to the stimuli.

## 2.1 Stimuli

Two simple rotational gestures were chosen to serve as stimuli for participants' performances; a slow, single rotation, and two rotations performed at a moderate and steady speed. These gestures were recorded for both the

acoustic and digital wind-like sounds. Another recording of a natural wind sound consisting of several short gusts of varying speed was chosen from the BBC Sound Effects Library [20] for use in the practice step.

## 2.2 Apparatus

The evaluation took place in an acoustically treated room at the Department of Theatre, Film and Television at the University of York. A laptop running the python-based Open Sesame experiment platform [21] presented questions and collected data from participants. A second laptop was used to run the prototype digital model in Max/MSP, and an additional computer was set up to deliver the sound stimuli and record participants' performances using Pro Tools. Both the Max/MSP patch and the Pro Tools session were obscured from participants to ensure they did not receive any additional visual feedback during their performances.

The sound stimuli and live audio of participants' performances was delivered to them via Pro Tools through a closed-back pair of Sennheiser HD280 Pro headphones. Participants' performances in response to the sound stimuli were recorded into the same Pro Tools session. The acoustic wind machine was obscured, apart from its crank handle, behind a cardboard screen to ensure that it provided no visual feedback to participants during performance (Figure 3).

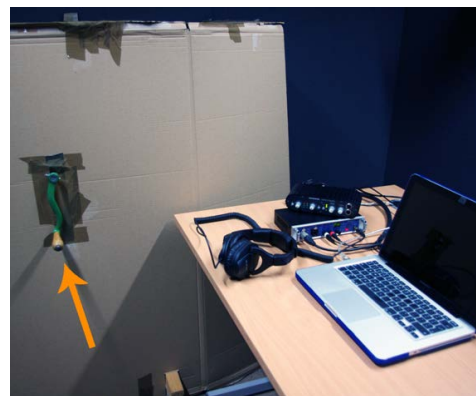


Figure 3. The experimental setup with crank handle highlighted.

## 2.3 Participants

The evaluation was undertaken with 48 participants. Of these, 32 identified themselves as female and 16 as male. 38 participants designated themselves as 18-24, 8 as 25-34, and 2 as 45-54 years old. 13 participants said they did not have experience of playing a musical instrument, 15 played a musical instrument at beginner level, 12 at intermediate level and 8 at advanced level. All participants reported normal hearing and were paid for their participation.

## 2.4 Procedure

The evaluation was based on a repeated measures design, with all participants performing with both the acoustic and



digital wind-like sounds in response to all of the stimuli. To avoid order effects, the order of presentation of the acoustic and digital wind-like sounds was randomised. The order of presentation of the sound stimuli was also randomised. This created four groups of twelve participants. Each group had its own order of system performed and stimuli presented (Table 1).

First System Performed	Subgroup	First Stimuli Presented
Acoustic	A	Acoustic
	B	Digital
Digital	A	Acoustic
	B	Digital

Table 1. The different orders of system and stimuli for this evaluation.

Participants were presented with the crank handle and advised that they would be able to perform a wind sound by rotating it. They were told that there would be two wind sounds to perform with during this evaluation, and that they would get to perform with both of these sounds, one after the other. No terms such as ‘acoustic’ or ‘digital’ were used to ensure that participants’ responses would not be influenced. Participants were then asked to listen to a wind sound from the group of stimuli played through their headphones, and then try to imitate what they had heard directly afterwards by turning the crank handle. There was a practice step, and then a test step, for both the acoustic and digital wind-like sounds. During each practice step, participants imitated the natural wind sound [20] and answered all of the questions that would be presented during the test step.

Participants were presented with a range of test questions to evaluate their experiences. They were first asked to rate how similar they perceived their own performances to be to the stimuli on a scale of 1 (*not similar at all*) to 7 (*as similar as they can possibly be*). Participants were then asked to rate how far they agreed with the statement “This wind sound is easy to play” on a scale of 1 (*strongly disagree*) to 7 (*strongly agree*).

Next, a list of possible descriptors for the wind-like sound that had been performed was presented, and participants were asked to describe the wind sound they had just played by selecting from these. There was also a space to add a descriptor of their own to this list. Finally, participants were given the opportunity to provide some free description of their experiences of playing each of the wind-like sounds.

### 3. RESULTS AND ANALYSIS

#### 3.1 Perceived Similarity of Performances to Stimuli

Participants’ ratings of perceived similarity between the sound stimuli and the wind-like sounds they had performed to imitate them were scored with values from 1 to 7. A Kruskal-Wallis test was then performed on the similarity ratings given by the participants across each of the groups

according to the order of performance system and the order of presentation of stimuli. This test confirmed that there was no statistically significant difference between the ratings given according to the experimental condition, confirming that no order effects had influenced the ratings (Table 2).

Test: Kruskal-Wallis	Significance	Effect Size
Acoustic similarity H(3) = 6.36	p > 0.05 not significant	-0.12 (small) power = 0.8
Digital similarity H(3) = 3.04	p > 0.05 not significant	0.0 (no effect) power = 0.8

Table 2. Results of the statistical testing to confirm no order effects influenced the similarity ratings for the acoustic or digital wind-like sounds.

A summary of the similarity ratings showed that, while there was a range of scores for each of the interactions, the acoustic wind machine performances had a higher mean rating for similarity to the stimuli presented than the prototype digital wind machine performances (Table 3).

Sound Played	Mean	SD	Median
Acoustic	4.88	1.66	5.5
Digital	2.77	1.51	2.5

Table 3. Summary of ratings for the acoustic and digital wind-like sounds’ similarity to the stimuli.

A Wilcoxon signed rank test was then performed on these similarity ratings, which confirmed that there was a statistically significant difference between the ratings given to the acoustic wind machine performances and the performances with its digital counterpart (Table 4). Participants therefore rated the similarity of the wind machine performances to the stimuli significantly differently depending on whether they were performing an acoustic or digital wind-like sound.

Test: Wilcoxon Signed-Rank	Significance	Effect Size
Z = -5.40	p < 0.01	-0.78 (large) power = 0.8

Table 4. Results of the statistical testing of participants’ similarity ratings.

#### 3.2 Perceived Easiness of Play

Participants’ scores for their responses to the statement “This wind sound is easy to play” were scored with values from 1 to 7. A Kruskal-Wallis test was then performed on these ratings given across each of the groups according to the order of performance system and the order of presentation of stimuli. Again, this confirmed that there was no statistically significant difference between

the ratings given according to each experimental condition, confirming that no order effects had influenced the results (Table 5).

Test: Kruskal-Wallis	Significance	Effect Size
Acoustic similarity H(3) = 5.36	p > 0.05 not significant	0.03 (small) power = 0.8
Digital similarity H(3) = 1.33	p > 0.05 not significant	0.0 (no effect) power = 0.8

Table 5. Results of the statistical testing to confirm no order effects influenced the easiness ratings for the acoustic or digital wind-like sounds.

A summary of the easiness ratings showed that the acoustic wind machine had a higher mean rating for ease of play than the prototype digital wind machine (Table 6).

Sound Played	Mean	SD	Median
Acoustic	4.98	1.19	5
Digital	3.04	1.41	3

Table 6. Summary of ratings for the acoustic and digital wind-like sounds' ease of play.

A Wilcoxon signed rank test was then performed on the easiness ratings to statistically compare the results for each wind-like sound. This test confirmed a statistically significant difference between how easy the acoustic and digital wind-like sounds were perceived to play (Table 7). Participants therefore rated the acoustic wind machine as significantly easier to perform with than its digital counterpart.

Test: Wilcoxon Signed-Rank	Significance	Effect Size
Z = -5.62	p < 0.01	-0.81 (large) power = 0.8

Table 7. Results of the statistical testing of participants' easiness ratings.

### 3.3 Descriptions of Sounds

Participants were then invited to describe the acoustic and digital wind-like sounds by choosing as many descriptors as they liked from a list. These descriptors were associated with a range of categories, including weather (*breeze*, *gale*), force (*gentle*, *strong*), onomatopoeic descriptions of wind (*shrieking*, *howling*), and a historical action-oriented onomatopoeic descriptor (*swishing* [1]).

Participants' responses to this question were collated to produce a bar graph in R comparing the frequency of the descriptors given to each wind machine (Figure 4). Participants chose not to add their own descriptors to the list, but instead chose from the descriptors provided.

This showed that the most popular descriptor for both the acoustic and digital wind-like sounds was the

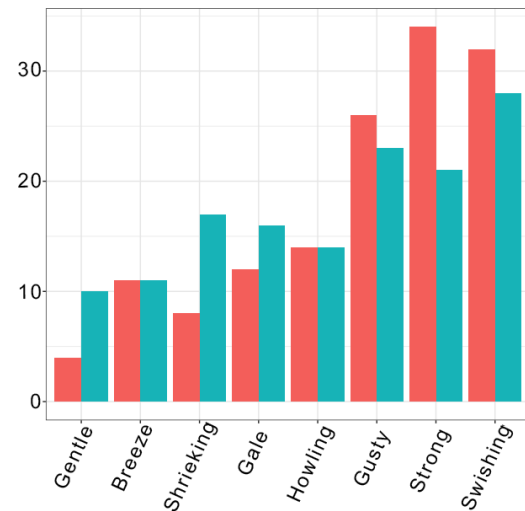


Figure 4. Summary of the descriptors participants assigned to their performances of the acoustic (blue) and digital (red) wind-like sounds.

action-oriented *swishing*, followed by the force descriptor *strong* and the weather-associated *gusty*. The acoustic wind machine scored more highly across these three descriptors than its digital counterpart. The digital wind-like sound was described with a fuller spread of adjectives, and was described more often as *shrieking* and *gale* when compared with its acoustic counterpart. This may reflect the fact that participants perceived the digital wind-like sound as having a narrower bandwidth of frequencies than the acoustic wind-like sound in performance.

### 3.4 Free Descriptions

The free descriptions participants gave of their experiences of performing with the acoustic and digital wind-like sounds were collated and coded. It was evident that participants had acquired some vocabulary from the list of descriptive words previously presented to them, as words like *gentle*, *strong* or *gusty* were included within their free descriptions. Some interesting issues and trends emerged. Participants readily connected the speed of rotation of the handle with what they variously described as the speed, motion, rhythm or pace of the resulting wind-like sound, whether it was acoustic or digital in origin. Some participants reported that the crank handle felt heavier to turn when performing the acoustic wind-like sound. One participant highlighted that they perceived a disconnection between the crank handle movement and the digital wind-like sound. Despite being informed that they would be playing wind sounds with the crank handle, one participant identified the digital wind-like sound as a rain sound in their comments.

### 3.5 Acoustic Analysis of Performed Sounds

The evaluation produced a corpus of recordings of participants' performances of the acoustic and digital

wind-like sounds in response to both the acoustic and digital stimuli. These recordings were exported from Pro Tools as audio clips and coded for analysis according to the performance gesture (a single slow rotation or two steady rotations) and the sound being performed (acoustic or digital). The coded audio clips were then analysed in Matlab using the MIR Toolbox [22] to produce numerical measures of the spectrum (brightness, inharmonicity, spectral centroid, spread and skewness) and amplitude envelope (event density - a measure of the frequency of onsets). The resulting numerical values for each feature were then collated together for statistical analysis in R.

To establish whether the source of the stimulus presented to participants (acoustic or digital) might have influenced their performances, gestures performed with the same system were paired in order to facilitate their statistical comparison. For example, two rotations performed with the acoustic wind machine in response to an acoustic stimulus were compared to two rotations performed with the acoustic wind machine in response to a digital stimulus. A Wilcoxon signed rank test was then performed to compare each acoustic feature of the paired gestures. This testing established that no statistically significant difference existed across the spectral measurements of the performances. For the measures of event density, no statistically significant difference was found between the paired gestures of two steady rotations. However, statistically significant differences were found for measures of event density for a single rotation performed with both the acoustic wind machine and the prototype digital wind machine (Table 8).

This suggests that the gesture of two rotations performed with the acoustic and digital wind-like sound was quite consistent regardless of whether participants had first listened to a stimulus that matched the sound that they were performing. For a single rotation, performances seem to have been more directly influenced by whether the stimulus presented matched the sound of the wind machine being played.

Test: Wilcoxon Signed-Rank	Significance	Effect Size
Acoustic Wind Event Density (1 rotation) $Z = -3.46$	$p < 0.01$	-0.49 (medium) power = 0.8
Digital Wind Event Density (1 rotation) $Z = -2.14$	$p < 0.05$	0.3 (medium) power = 0.8

Table 8. Results of the statistical testing to compare the acoustical analysis of participants' performances.

#### 4. DISCUSSION

This evaluation aimed to establish whether there was perceived similarity between the experience of performing

with the acoustic wind machine and that of performing with its digital counterpart. The results established that, while the continuous sonic feedback was the only mode of feedback that changed between these two performance conditions, participants found the acoustic wind machine significantly easier to play and perceived it as sonically similar to the stimuli used to elicit their performances. By contrast, the digital wind-like sound was rated as significantly less easy to play, and participants found their performances with it to be significantly less similar to the stimuli they were trying to imitate. Statistical testing showed that the ratings for similarity and ease of play were significantly different depending on the kind of wind-like sound being rated, and so the results did not allow the null hypothesis to be rejected. These results suggest that the digital model of the acoustic wind machine needs to be developed further. In particular, the easiness ratings for the digital wind-like sounds may reflect the need to improve the model's response to variations in performance gesture. Participants may have experienced this as an action-sound latency issue, something which previous research has shown to be disruptive to musical performance [23].

When asked to choose from a list of descriptors for the acoustic and digital wind-like sounds, participants preferred the action-based descriptor *swishing* for both sounds, and were more confident in categorising the acoustic wind machine (as *gusty*, *strong* and *swishing*). Some interesting information emerged from participants' free description of their performances, in particular that the change in sonic feedback from an acoustic to digital sound might have influenced how the physical properties of the acoustic wind machine were experienced. This concurs with previous research showing that auditory cues can influence the perception of haptics and movement [24–26]. This aspect of the change in sonic feedback from an acoustic to digital wind-like sound could be explored further in a future evaluation.

Acoustical analysis of the corpus of wind sounds produced from recordings of participants' performances established that there was no statistically significant difference in the acoustical measurements of sounds performed in response to a stimulus that matched the wind-like sound being played when compared with performances responding to an unmatched stimulus. The exception to this finding was the measurement of event density, or number of onsets in the sound's amplitude envelope per second, which was found to be significantly different for a single rotation performed with the acoustic wind machine between the acoustic and digital wind stimuli. The same pattern was visible for a single rotation with the digital wind-like sound.

This suggests that participants played the wind-like sounds quite differently depending on the kind of stimulus (acoustic or digital) presented to them to elicit their performance. However, this difference was not evident in the gestures of two steady rotations. It is possible that participants understood the stimuli of two steady rotations much more easily, but given the lower ratings for similarity and easiness participants gave to the digital

wind-like sound, it is unlikely that the digital stimuli were so simple to imitate. It is proposed that this continuity of gestural response evidenced in the performances of two steady rotations may be the result of the mechanical qualities of the acoustic wind machine itself, rather than the responses of participants. With a single rotation, the acoustic wind machine's cylinder may not have time to accumulate rotational energy and push forward from the movement of the performer's hand on the crank handle. However, with a gesture of two rotations, the moving cylinder must be imposing more of its flywheel qualities, and hence some regularity, on the performer's rotational movement. Given the medium effect size observed here, further testing with a larger number of participants would be able to confirm these results. An experiment examining a broader range of gestures, and in particular a robust method of recording data from the rotary encoder would help to illustrate the influence of the cylinder's rotational inertia on the performer's movement in the continuous sonic interaction.

## 5. CONCLUSION

The evaluation of the acoustic wind machine and its digital counterpart in performance has confirmed that the sonic response of the digital model is not yet perceptually close enough to the acoustic wind-like sound to be used as a substitute for it in a future experiment. Further work is therefore needed to calibrate the response of the digital model. However, the acoustic wind machine was itself rated highly for ease of performance and similarity to the stimuli it imitated, confirming its enactive qualities. The potential of the mechanical wooden interface playing a role in facilitating a meaningful link between a performer's action and the complex wind-like sound is interesting, as the flywheel properties of the cylinder and axle design may have a critical role in enhancing the enactive potential of this particular theatre sound effect design. Isolating the sonic feedback as part of this evaluation has also shown that despite the continuity of tactile and kinaesthetic feedback across the interactions, participants perceived their acoustic and digital performances significantly differently.

Using historical theatre sound effect designs as the focus of an evaluation like this allows participants' perceptual experiences of incrementally different modes of feedback, in a continuous sonic interaction, to be explored in detail. How far the digital model needs to be developed in order to capture more of the enactive experience of the acoustic wind machine in performance should be investigated. In this way, the potential of digital systems to afford rich, intuitive encounters with performable everyday sounds can be explored further.

## Acknowledgments

This research was supported by the Arts and Humanities Research Council (AHRC) through the White Rose College of the Arts and Humanities Doctoral Training Partnership (WRoCAH).

## 6. REFERENCES

- [1] F. Keenan and S. Pauleto, "Listening back: Exploring the sonic interactions at the heart of historical sound effects performance," *The New Soundtrack*, vol. 7, no. 1, pp. 15–30, 2017.
- [2] M. Chion, "Epilogue. audition and ergo-audition: Then and now," in *See this Sound: Audiovisuality: a Reader*, D. Daniels and S. Naumann, Eds. Buchhandlung Walther König, 2015, pp. 670–684.
- [3] K. Franinović, "Amplified movements: An enactive approach to sound in interaction design," *New Realities: Being Syncretic*, pp. 114–117, 2009.
- [4] —, "Amplifying actions - towards enactive sound design," PhD Thesis, 2013.
- [5] W. W. Gaver, "What in the world do we hear?: An ecological approach to auditory event perception," *Ecological psychology*, vol. 5, no. 1, pp. 1–29, 1993.
- [6] C. Poepel, *On interface expressivity: a player-based study*. National University of Singapore, 2005.
- [7] E. R. Miranda and M. M. Wanderley, *New digital musical instruments: control and interaction beyond the keyboard*. AR Editions, Inc., 2006, vol. 21.
- [8] A. R. Jensenius and M. J. Lyons, *A NIME Reader: Fifteen years of new interfaces for musical expression*. Springer, 2017.
- [9] B. L. Giordano, Y. Visell, H.-Y. Yao, V. Hayward, J. R. Cooperstock, and S. McAdams, "Identification of walked-upon materials in auditory, kinesthetic, haptic, and audio-haptic conditions," *The Journal of the Acoustical Society of America*, vol. 131, no. 5, pp. 4002–4012, 2012.
- [10] E. Frid, J. Moll, R. Bresin, and E.-L. S. Pysander, "Haptic feedback combined with movement sonification using a friction sound improves task performance in a virtual throwing task," *Journal on Multimodal User Interfaces*, pp. 1–12, 2018.
- [11] S. Serafin and A. De Götzen, "An enactive approach to the preservation of musical instruments reconstructing russolo's intonarumori," *Journal of New Music Research*, vol. 38, no. 3, pp. 231–239, 2009.
- [12] A. Farnell, *Designing sound*. MIT Press Cambridge, 2010.
- [13] R. Selfridge, J. D. Reiss, E. J. Avital, and X. Tang, *Physically derived synthesis model of an Aeolian tone*. Audio Engineering Society, 2016.
- [14] R. Selfridge, D. Moffat, and J. D. Reiss, "Sound synthesis of objects swinging through air using physical models," *Applied Sciences*, vol. 7, no. 11, p. 1177 models, 2017.



- [15] J. Woodhouse, "Physical modeling of bowed strings," *Computer Music Journal*, vol. 16, no. 4, pp. 43–56, 1992.
- [16] S. Baldan, S. Delle Monache, and D. Rocchesso, "The sound design toolkit," *SoftwareX*, vol. 6, pp. 255–260, 2017.
- [17] J. O. Smith, *Physical audio signal processing: For virtual musical instruments and audio effects*. W3K Publishing, 2010. [Online]. Available: <https://ccrma.stanford.edu/~jos/pasp/>
- [18] F. Keenan and S. Pauletto, "An acoustic wind machine and its digital counterpart: Initial audio analysis and comparison," in *Interactive Audio Systems Symposium (IASS)*, University of York, York, UK, 2016. [Online]. Available: <http://www.york.ac.uk/sadie-project/IASS2016.html>
- [19] —, "Design and evaluation of a digital theatre wind machine," in *Proceedings of The 17th International Conference on New Interfaces for Musical Expression (NIME 17)*, Copenhagen, Denmark, 2017.
- [20] BBC, "Weather 1," CD, 1988.
- [21] S. Mathôt, D. Schreij, and J. Theeuwes, "Opensesame: An open-source, graphical experiment builder for the social sciences," *Behavior research methods*, vol. 44, no. 2, pp. 314–324, 2012.
- [22] O. Lartillot and P. Toivainen, "A matlab toolbox for musical feature extraction from audio," in *International Conference on Digital Audio Effects (DAFX)*, Bordeaux, France, 2007, pp. 237–244.
- [23] R. H. Jack, A. Mehrabi, T. Stockman, and A. McPherson, "Action-sound latency and the perceived quality of digital musical instruments: Comparing professional percussionists and amateur musicians," *Music Perception: An Interdisciplinary Journal*, vol. 36, no. 1, pp. 109–128, 2018.
- [24] D. E. DiFranco, G. L. Beauregard, and M. A. Srinivasan, "Effect of auditory cues on the haptic perception of stiffness in virtual environments," in *American Society of Mechanical Engineers, Dynamic Systems and Control Division (Publication) DSC*, vol. 61, 1997, pp. 17–22.
- [25] F. Avanzini and P. Crosato, "Haptic-auditory rendering and perception of contact stiffness," in *International Workshop on Haptic and Audio Interaction Design*. Springer, 2006, pp. 24–35.
- [26] L. Turchet, S. Serafin, and P. Cesari, "Walking pace affected by interactive sounds simulating stepping on different terrains," *ACM Transactions on Applied Perception (TAP)*, vol. 10, no. 4, p. 23, 2013.

# Adaptive Loudness Compensation in Music Listening

**Leonardo Fierro**

Signals and Communication Lab,  
Dept. Information Engineering,  
University of Brescia, Brescia, Italy  
lfuegofierro@gmail.com

**Jussi Rämö and Vesa Välimäki**

Acoustics Lab,  
Dept. Signal Processing and Acoustics,  
Aalto University, Espoo, Finland  
jussi.ramo@aalto.fi  
vesa.valimaki@aalto.fi

## ABSTRACT

The need for loudness compensation is a well known fact arising from the nonlinear behavior of human sound perception. Music and other sounds are mixed and mastered at a certain loudness level, usually louder than the level at which they are commonly played. This implies a change in the perceived spectral balance of the sound, which is largest in the low-frequency range. As the volume setting in music playing is decreased, a loudness compensation filter can be used to boost the bass appropriately, so that the low frequencies are still heard well and the perceived spectral balance is preserved. The present paper proposes a loudness compensation function derived from the standard equal-loudness-level contours and its implementation via a digital first-order shelving filter. Results of a formal listening test validate the accuracy of the proposed method.

## 1. INTRODUCTION

Loudness compensation is based on the equal-loudness-level contours first reported by Fletcher and Munson in the 1930s [1] and different approaches to loudness compensation have been discussed since then [2–5]. It is well known how perceived bass and sub-bass ranges are much more affected than high frequencies when the sound level goes down. As a consequence, it is beneficial to adapt the compensation based on the listening level of the audio track.

Recently, Prasad described a compensation based on and approximation of the difference in sensitivity, which can be implemented using a filterbank or fast convolution based on the FFT (Fast Fourier transform), which causes some processing latency [6]. Hawker and Wang proposed the use of a scalar function describing the change in SPL required to effect a change of 1 Phon. Their method is implemented using FIR (Finite Impulse Response) filters with 1024 coefficients, which are also best to implement using fast convolution [7].

According to Katz [8], music is nowadays usually mixed and mastered with loudspeakers at the sound pressure level (SPL) of 83 dB, or more generally at SPL between 80 to

85 dB. In such a range, human loudness perception is the closest to be flat while avoiding painful levels. However, those are quite high levels and a prolonged exposure can tire the listener or even damage the hearing [9] [10]. Safer listening levels for consumers (in particular using headphones) lie in the 60–75 dB SPL range.

Inevitably, when the sound reproduction level is changed, the perceived spectral balance is altered as well and the fidelity to the original master is lost. The ultimate goal of a loudness compensation method is not to provide the best subjective bass compensation according to the listener, but to recover such lost fidelity by regaining the spectral balance of the playback sound. Consumer audio equipment sometimes offered a “loudness” switch, whose action was merely a constant bass boost regardless of the playback level or a variable analog shelving filter control without calibration [4, 11]. More recent devices have removed this feature, leaving the user to manually change the volume controls.

This paper proposes a computationally efficient compensation technique using a first-order IIR (Infinite Impulse Response) digital filter to improve the listening experience, based on the equal-loudness-level contours (ELLC) provided by the ISO226:2003 standard [12]. The proposed method is highly accurate approximating the ELLC curves within  $\pm 1$  dB. Furthermore, the low-order IIR filter does not introduce practically any processing latency. This is similar to the best analog loudness control circuits with the addition that it allows level calibration.

The rest of this paper is organized as follows. Section 2 briefly illustrates the ELLC, reporting the contour function and data interpolation useful for the proposed compensation method described in Section 3. Section 4 is related to the filter design, and Section 5 to the optimization of filter parameters. Description and results of conducted listening tests are shown in Section 6. Section 7 concludes this paper.

## 2. EQUAL-LOUDNESS-LEVELS CONTOURS

The ISO226 standard is used as reference for the work described in this paper. The standard specifies the sound pressure levels of a pure tone, as function of frequency, perceived as equally loud by human listeners in free space [12]. Polynomial function for the contours is given by:

Copyright: © 2019 Leonardo Fierro et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

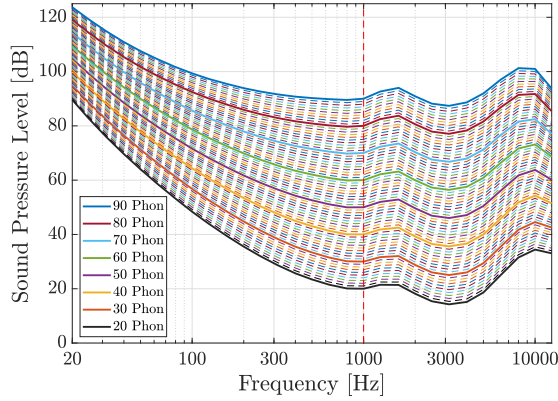


Figure 1: Interpolated ELLC for 20–90 Phon range between 20 Hz and 12.5 kHz.

$$L_p = \frac{10}{a_f} \log_{10} A_f - L_u + 94, \quad (1)$$

where

$$A_f = \frac{4.47}{10^3} \left( 10^{\frac{L_n}{40}} - 1.15 \right) + \left( 0.4 \cdot 10^{\frac{T_f + L_u}{10} - 9} \right)^{a_f}; \quad (2)$$

$L_p$  is the sound pressure level (in dB SPL) of a pure tone;  $L_n$  is the loudness level (in Phon);  $f$  is the frequency of the pure tone;  $T_f$  is the hearing threshold (in dB SPL);  $a_f$  is the exponential factor, accounting for loudness perception;  $L_u$  is the magnitude (in dB) of the frequency response, normalized at 1 kHz. The data range provided by the standard is 20 to 90 Phon, whereas the frequency spans from 20 Hz to 12.5 kHz, and it is shown in Fig. 1.

From the ELLC, it is easy to see how the sensitivity of human perception changes nonlinearly with frequency and how low-frequency range is the most heavily affected part of the spectrum. Data from ISO226 was linearly interpolated with 1-Phon steps to provide intermediate curves (Fig. 1).

### 3. COMPENSATION METHOD

The main idea behind the proposed method is quite straightforward: derive a *sensitivity function* from the ELLC, relative to the listening level, then find an inverse function to be used as a trace-guide for the design of a digital filter that can then correct the spectral balance.

It is possible to normalize each curve in Fig. 1, with respect to its SPL value at 1 kHz, in order to evaluate the sensitivity of human hearing for different SPLs, i.e. to obtain a sensitivity function  $S$ :

$$S(f, L_n) = -L_p(f, L_n) + L_p(1000, L_n). \quad (3)$$

For each sensitivity curve shown in Fig. 2, the difference in perception with respect to SPL at 1 kHz corresponds to the gain (or attenuation) to be introduced in order to have a flat response.

The mastering level ( $L_M$ ) and the listening level ( $L_L$ ) for music are usually different, as the latter is quieter than the

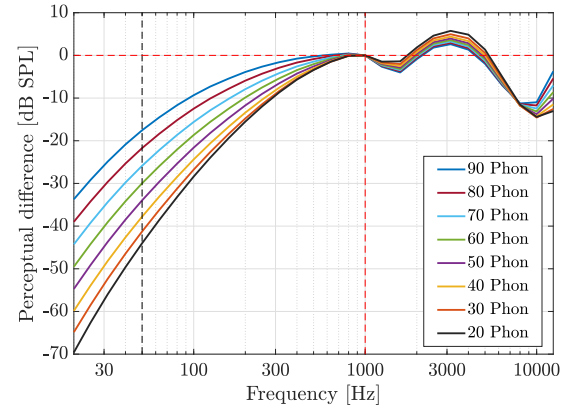


Figure 2: Sensitivity function, for different levels.

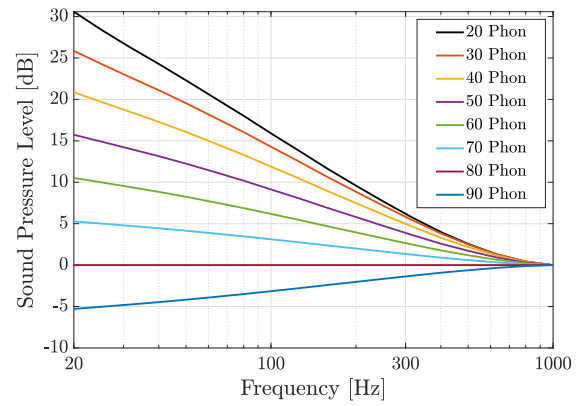


Figure 3: Derived filtering trace-guides at  $f \leq 1$  kHz for certain listening levels,  $L_M = 80$  dB SPL.

first one. The goal is to compensate the perceived spectral balance at  $L_L$  in such a way that it matches the perceived spectral balance at  $L_M$ . A relationship between the two levels is derived, identifying a *difference curve*  $\Delta L_p$ :

$$\Delta L_p(f, L_M, L_L) = L_p(f, L_M) - L_p(f, L_L) - N_f, \quad (4)$$

where  $N_f$  is the normalization factor for the sensitivity function, according to (2):

$$N_f = L_p(1000, L_M) - L_p(1000, L_L) = L_M - L_L. \quad (5)$$

Inverting (3), a balancing curve is finally obtained. It corresponds to the magnitude response that perfectly balances the perception of spectral components as intended by the mastering:

$$\begin{aligned} H(f, L_M, L_L) &= -\Delta L_p(f, L_M, L_L) \\ &= -L_p(f, L_M) + L_p(f, L_L) + N_f. \end{aligned} \quad (6)$$

Notice how bass reduction instead of boost is required for  $L_L > L_M$  in Fig. 3. This is significant for situations like live concerts or discos, where music can be played at high levels [11].

#### 4. FILTER DESIGN

Since a set of curves—referred as *trace-guide* from here on—has been obtained, the next step is to identify a type of digital filter whose magnitude response is sufficiently close to the trace-guide and that can easily adapt to a change in listening level; low order and low complexity are desired, in order to have minimum impact on the reproduction system and allow a real-time implementation.

IIR filters are a natural choice in terms of efficiency for many audio DSP applications [13]. Their processing is typically low demanding in terms of operations and memory, enabling the implementation of such filters in low cost architectures and products hitting the markets. FIR filters allow linear-phase processing, but require generally a larger number of operations per output sample and more memory on the DSP interface than IIR filters. Furthermore, FIR filters can be limited in resolution when working with low frequencies, affecting the quality of the filter coefficients [13]. For those reasons, FIR filters are not considered in this paper.

Digital filters can be derived from analog filters, converting a transfer function with analog poles/zeros from the Laplace-domain to the z-domain and obtaining a difference equation, using common transformations such as the bilinear transform, the matched Z-transform, the pole-zero mapping method, or the impulse-invariant method [14]. Digital filters can also be designed by choosing appropriate locations for the poles and zeros on the unit circle in the z-domain, thus imposing desired corner frequencies and slope [15].

Depending on the quality of the back-end application, IIR filters can be applied on both fixed-point and floating-point architectures. Consequently, either Direct Form I, II or Transpose II can be chosen for implementation: DF2 and DF2T require less delay elements but are prone to overflows with high-order filters, so DF1 may be a simpler and better solution for fixed-point architectures [15].

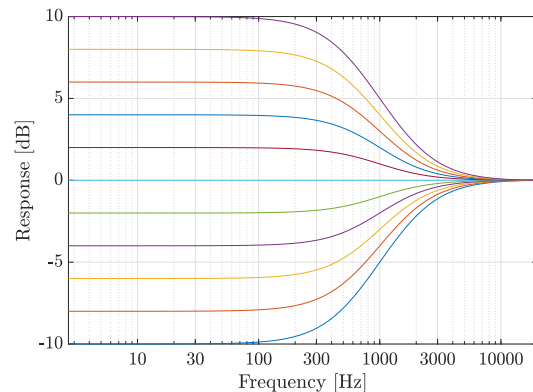
##### 4.1 Fractional order filters

From Fig. 3, it can be observed that the target magnitude response slope is less than 20 dB/decade, for every curve. This suggests to look for filters with an order smaller than one—the so called *fractional order filters*—and with a low-pass behaviour.

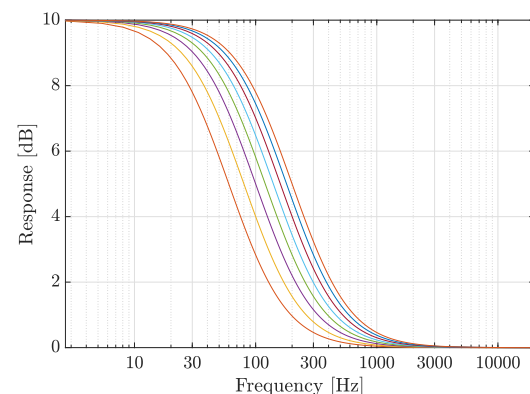
Fractional order filters (FOF) are neither well documented nor well described in DSP literature at the moment of writing [16], but a design process for digital FOFs has been proposed by Nielsen [11]. Although providing similar results as shelving filters (Section 4.2), the latter should eventually be preferred to FOFs, due to the wider use and lower complexity (always a good thing when discussing real-time audio applications). For this reason, FOFs are not considered in this paper. Future studies might provide interesting results, also relevant for this work.

##### 4.2 Shelving filters

A shelving filter boosts or attenuates the magnitude of an input signal in a certain frequency band—either the lowest



(a) Variable  $G$ , constant  $\omega_c = 1$  kHz



(b) Variable  $\omega_c$ , constant  $G = 10$  dB

Figure 4: Effects of gain and crossover frequency on the magnitude response of a first-order low shelving filter.

frequency band or the highest frequency band—without cutting out the harmonics in that band as a typical low-pass/high-pass filter would do [13, 17].

Depending on whether it affects the bass or the *treble* (high frequencies), it will be referred to as either a *low-shelving* or *high-shelving* filter, respectively.

This type of filter is largely used in parametric equalizers, due to the smooth transition of the response between affected and unaffected regions and the simple implementation. A classic parametric equalizer presents two knobs to the user, one for bass and one for treble<sup>1</sup>, through which it is possible to alter the filter shape and its effect on the playback sound.

Simplicity comes from the fact that the behavior of a shelving filter is completely described by just the gain  $G$  and the crossover frequency  $f_c$  (often also called *corner* or *cut-off* frequency). As can be seen from Fig. 4, the gain parameter affects the gain at low frequencies and the slope (Fig. 4a), while the crossover frequency parameter affects the width of the response, i.e. its frequency span (Fig. 4b).

Transfer functions for both the first and second-order low-shelving digital filters have been derived by Välimäki and Reiss [13]. The transfer function of the first-order shelf can

<sup>1</sup> Typically, one or more knobs adding peaks/notches in the mid-frequencies range are also available in common music equipment.



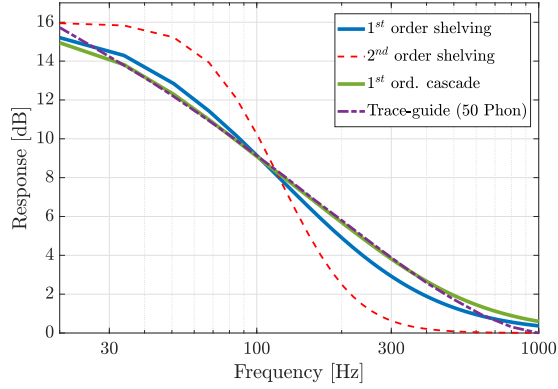


Figure 5: Comparison between trace-guide and shelving filters,  $L_M = 80$  dB SPL,  $L_L = 50$  dB SPL.

be written as

$$H_{LS}(z) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}}, \quad (7)$$

where

$$b_0 = \frac{G\Omega + \sqrt{G}}{\Omega + \sqrt{G}}, \quad b_1 = \frac{G\Omega - \sqrt{G}}{\Omega + \sqrt{G}}, \quad a_1 = \frac{\Omega - \sqrt{G}}{\Omega + \sqrt{G}},$$

$\Omega = \tan(\omega_c/2)$ , and  $\omega_c = 2\pi f_c/f_s$ .

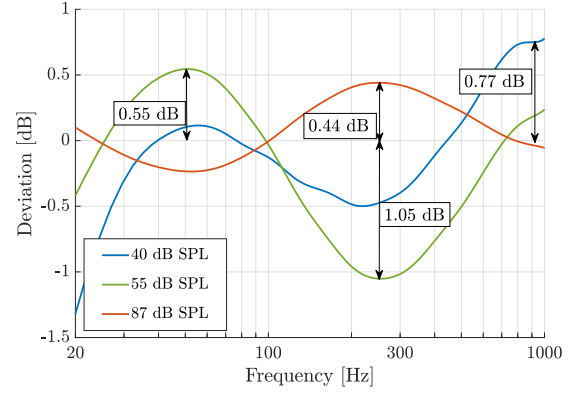
Fig. 5 shows a comparison of different low-shelving filter responses, where the trace-guide is interpolated with a spline function to provide more frequency points. As can be seen, the first-order shelving filter presents a fairly good approximation of the trace-guide; an even better result is achieved with a cascade of two first-order filters, but a second-order low-shelving returns curves which are too steep. Moreover, the flat response towards the lowest frequencies avoids unnecessarily boosting the frequencies close to DC, or 0 Hz.

## 5. OPTIMIZATION OF FILTER PARAMETERS

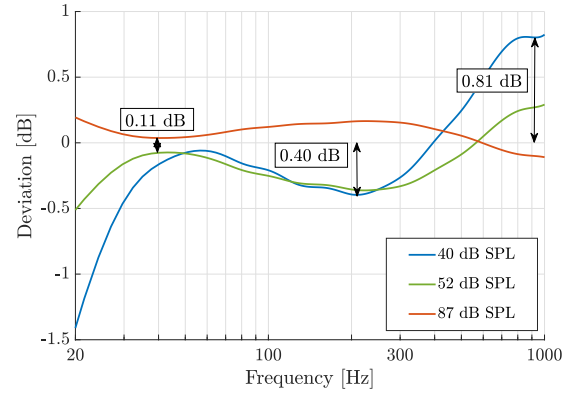
After choosing the filter type, the optimal parameters ( $G$ ,  $\omega_c$ ) should be found. In case of a cascade of two first-order shelves, there are four parameters: ( $G_1$ ,  $G_2$ ,  $\omega_{c1}$ ,  $\omega_{c2}$ ). However, few things might be taken into account in order to simplify the optimization problem:

- $L_M$  typically lies in a very limited interval (80–85 dB SPL), so trace-guides will be similar for levels in such a range;
- it is reasonable to choose trace-guide SPL at 20 Hz as  $G$ ; in case of a filter cascade, the product of the gains (the sum, in the log domain) should match such value.

Holding to these considerations, an optimization algorithm can be run to identify the optimal parameters. A genetic algorithm (GA) has been chosen for this task [18], due to its suitability to solving search problems and the high-quality solutions it is capable of generating in a reasonable time.



(a) Single first-order filter



(b) Cascade of first-order filters

Figure 6: Worst-case deviations from trace-guide given by first-order low-shelving filters, with peaks of max deviation.

## 5.1 Crossover frequencies

Initial GA runs over the 80–85 dB SPL range for  $L_M$  show that the optimal solution for gains in the shelving cascade is really close to an equal weighting. So it is safe to assume, in first approximation:

$$G_1 = G_2 = \frac{1}{2} G. \quad (8)$$

This way, the complexity of the optimization task has already been reduced by one degree. Of course, this does not concern the single filter case.

Then, the crucial step in optimization seems to be the choice of the poles, i.e. the crossover frequencies. It is easy to change filter parameters in real-time application. However, given the short range of considered mastering levels and the definite frequency span of the trace-guide, fixing the poles simplifies the problem even further without loss of generality, leaving only  $G$  to be modified as  $L_L$  changes.

Multiple GA runs return, as consistent optimal solution:  $f_c = 122$  Hz for the first-order shelving filter; and  $f_{c1} = 61.1$  Hz and  $f_{c2} = 242$  Hz for the cascade of first-order shelving filters. Maximum deviations from trace-guide are plotted in Fig. 6. As can be seen, both cases provide inter-

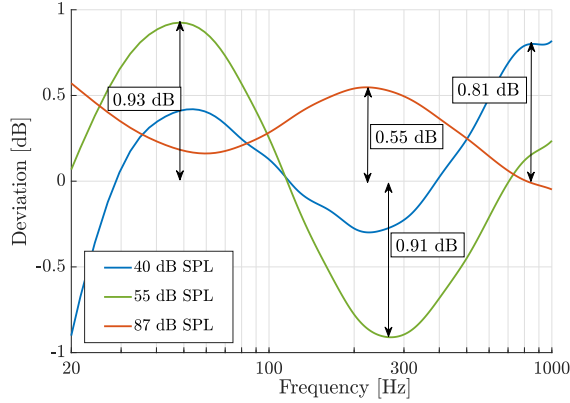


Figure 7: Deviation from trace-guide given by first-order low-shelving filter, with gain adjustment  $\alpha = 0.485$  dB.

esting results, with small errors for all considered listening levels and minimum error on the mid frequencies. To achieve high-fidelity, a maximum deviation of  $\pm 1$  dB from the trace-guide is desired. As shown in Fig. 6b, the filter cascade error always lies inside such a range, while the single filter deviation slightly exceeds  $-1$  dB around 250 Hz (Fig. 6a). Although, the cascade already satisfies requirements, the use of a single filter is desirable to further reduce complexity and computation time.

## 5.2 Gain adjustment

Since the crossover frequency has been fixed, a possible solution is to adjust the gain with a small bias term  $\alpha$  in order to compensate for the deviation peak around 250 Hz, without exceeding the range somewhere else. The modified filter gain is then determined as:

$$G_{dB} = \Delta L_p(20, L_M, L_L) + \alpha. \quad (9)$$

Running the GA again, an optimal bias term  $\alpha = 0.485$  dB was found. Fig. 7 shows that this small bias reduces the maximum deviation, while maintaining the error between  $\pm 1$  dB in the rest of the bass range.

Fig. 8 shows the magnitude responses of the first-order shelving filters using the modified gain and (7). The filter coefficients used for these curves are listed in Table 1.

## 6. LISTENING TEST

### 6.1 Design

A listening test was conducted on a selection of experienced listeners. No one reported any hearing impairments or medical conditions. The test was designed for this purpose and conducted in the MATLAB environment on a MacOS computer, using a pair of Sennheiser HD 650 dispatched inside a listening booth at the Aalto Acoustics Lab.

Audio samples were chosen from different genres for having a prominent bass line and other different spectral features. They consisted of short tracks (4 to 8 seconds) cut from the following songs:

1. Queen, “Another One Bites The Dust” (1980);

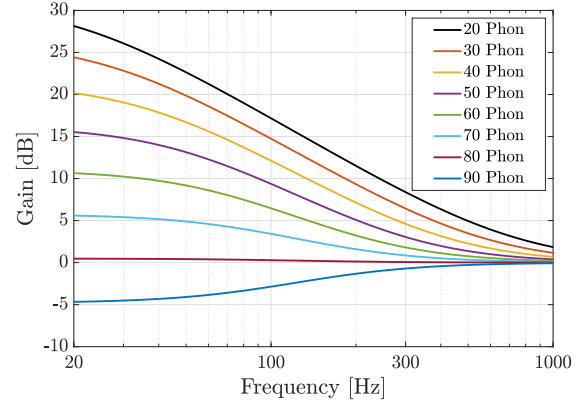


Figure 8: Magnitude responses of the first-order shelving filter at  $f \leq 1$  kHz for certain listening levels, when  $M_L = 80$  dB. Cf. Fig. 3.

$L_L$ [dB]	Numerator		Denominator
	$b_0$	$b_1$	$a_1$
90	0.9952	-0.9821	-0.9773
80	1.0005	-0.9827	-0.9832
70	1.0058	-0.9818	-0.9876
60	1.0117	-0.9791	-0.9908
50	1.0186	-0.9746	-0.9932
40	1.0271	-0.9678	-0.9949

Table 1: Shelving filter coefficients for various choices of  $L_L$ , when  $M_L = 80$  dB.

2. White Stripes, “Seven Nation Army” (2003);
3. Daft Punk, “Around The World” (1997).

From further on, each track will be identified with its number from the list above, e.g. Track 1, Track 2, Track 3. Track 2 is composed by just bassline and drumline, showing narrow spectral content concentrated in the bass range. Track 3 present a broader spectral content; same for Track 1, which also includes vocals.

Subjects under test were presented with 7 instances of each track, for a total of 21 stimuli pairs. Each step held a version of the track played at  $L_M$  (see Section 6.2), named *reference*, and an attenuated variant. Applied loudness reduction varied between 0 and 40 dB in steps of 10 dB, corresponding to five different listening levels in the 40–80 dB SPL range. 80 dB SPL (no reduction, same as reference) and 60 dB SPL (20 dB attenuation) were presented twice per each track: repeated reproductions were used during the screening phase to evaluate subject consistency and then discarded before statistical analysis of results.

Loudness compensation was applied to the attenuated variant using the single first-order low-shelving filter. The crossover frequency was fixed at 122 Hz (see Section 5.1), and the subjects modified the filter gain using a slider during the test. The slider selected a different gain for the filter based on the ELLC trace-guide (Section 5). Slider movements were discretized and each step corresponded

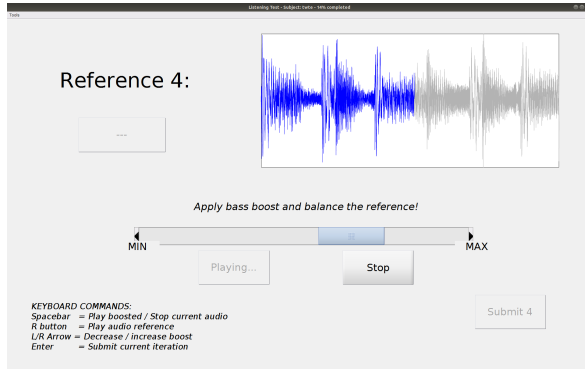


Figure 9: Screen-shot of the test GUI.

to a 2-dB variation in the selection of the curve.

The subjects were asked to focus on the bass of the played sounds, in particular on the balance between the overall loudness and the bass loudness of the reference, and to compensate the spectral balance of the variant in order to match the reference balance, but at a different  $L_L$ . Subjects had access to a horizontal slider (Fig. 9) that, it was told them, allowed to give “*boost or reduction*” to the bass of the variant. The range of the slider was hidden and slightly randomized, having only the labels “*Min*” and “*Max*” at its two extremes.

The audio samples were set to play in a continuous loop until they were manually stopped. While it was possible to reproduce the variant as many times and as long as as desired, the play count of the reference was limited to two, the first starting automatically at each new step of the test. This means that, after stopping the reference the first time, it was possible to play it again just one more time. This choice was made to avoid the listener to go “back and forth” from the reference to the variant and force them to pay extra focus on the task.

Given the fast decay of human memory of sounds, the subjects have been suggested to get a general idea of the frequency components of the reference during the first play, then to reproduce the variant and explore the amount of possible “boost” given by the slider, before getting back to the reference and gain a more clear sense of the spectral balance. After the second stop, a final choice for variant compensation should have been made.

Listeners were allowed a short training session before starting the actual test to get acquainted with the interface, the keyboard shortcuts and the task itself. The results of the training session were not included in the statistical analysis.

## 6.2 Level calibration

Having an accurate measurement of the loudness level was critical for the goodness of the test, so a calibration phase was performed. Used instrumentation involved a RME Fireface 800 and a G.R.A.S. 45CA Headphone Test Fixture in compliance with the IEC 60318-4:2010 occluded-ear simulation [19].

The different tracks, played through the headphones allocated on the ear simulator, were loudness matched by using

	Loudness [LUFS]	Max SPL [dBA]
<b>Track 1</b>	−11.3	83.4
<b>Track 2</b>	−11.1	77.0
<b>Track 3</b>	−11.0	81.2

Table 2: ITU-based loudness in loudness units relative to full scale (LUFS) according to ITU-R BS.1770-4 and the maximum measured SPLs (in dBA).

the ITU-R BS.1770-4 [20] loudness measure. Their playback level was then set to be close to 80 dBA (A-weighted dB). The actual measured dBA vary, since the levels depend on the contents of the considered track. The ITU-based loudness levels and maximum A-weighted dB levels are reported in Table 2.

## 6.3 Screening

In order to isolate inconsistent listeners, 80 dB SPL and 60 dB SPL cases were presented twice. For screening, it was not tested how accurate subjects were, but their degree of repeatability. For this reason, the absolute difference between the first and the repeated value was calculated at both levels, for each song and for every subject. A double-threshold method was implemented to evaluate consistency:

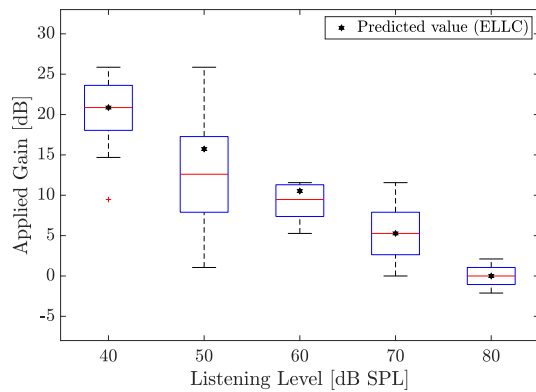
1. If  $\forall i \Delta 80_i \leq 6$  dB, subject is consistent;
2. Otherwise, if  $\Delta 80_i > 6$  dB for one track and  $\Delta 60_i \leq 10$  dB for at least two tracks, subject is consistent;
3. Otherwise, subject is inconsistent and discarded.

Here  $i = 1, 2, 3$  is the number of the track and  $\Delta 80_i$  and  $\Delta 60_i$  are the differences of the two instances. Since human perception was evaluated, it was reasonable to have a stricter threshold at the reference level (80 dB), where the spectral balance of two signals with the same level were matched, and a more relaxed threshold for the attenuated level (60 dB), which required a harder task of matching the spectral balance of two signals with different listening levels. A total of 18 subjects participated in the test; 11 of them passed the consistency screening and were included in the analysis.

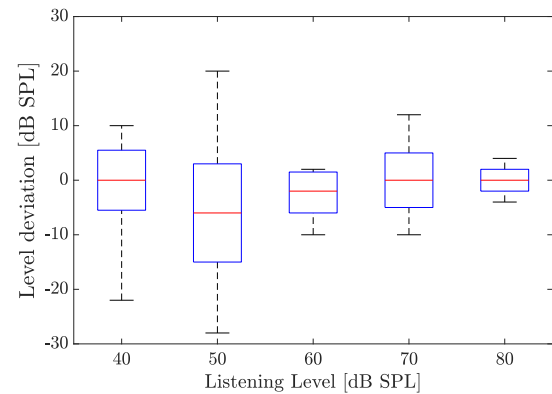
## 6.4 Results

Results from the listening test show that ELLC can reproduce the average response of the listeners. This is shown by the box plots in Figs. 10 and 11, where the box plots present the median (red line), the 25<sup>th</sup> and 75<sup>th</sup> percentiles (blue rectangle), the extension to the most extreme data points not considered outliers (black whiskers) and the outliers (red cross). Furthermore, the black markers represent the predicted correct compensation that matches the level of the ELLC.

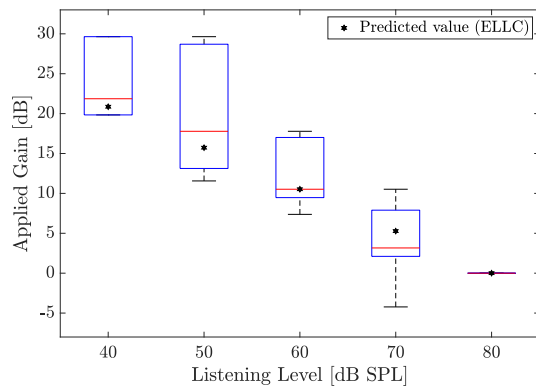
Fig. 10 plots the applied compensation versus the listening level of the track, showing how such compensation adapts to the level and increases towards the lowest levels.



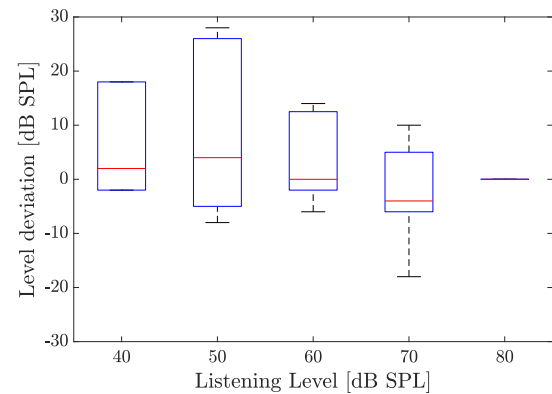
(a) Track 1



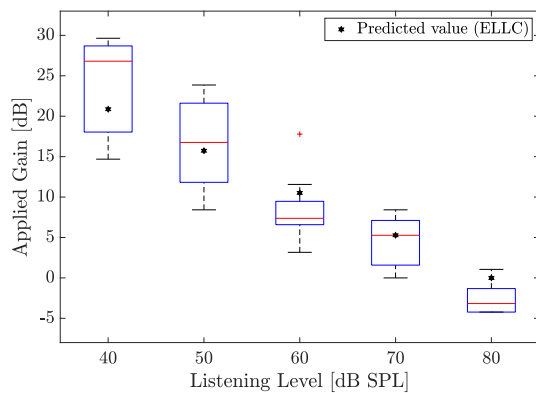
(a) Track 1



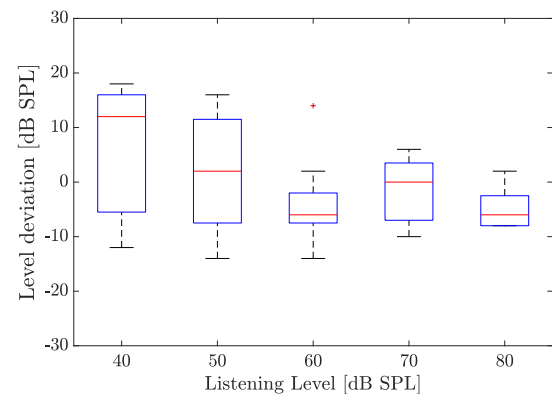
(b) Track 2



(b) Track 2



(c) Track 3



(c) Track 3

Figure 10: Test results, grouped by song. Box-plot of compensation introduced by subjects at each listening level.

Fig. 11 shows the level deviations, so it is easier to see the goodness of the results and the cases of under or over compensation. As expected, data presents moderate variance, due to the difficulty of the task; nevertheless, the median of the error in level evaluation always lies in a close range near 0 dB (Fig. 11).

Analyzing the results, it is possible to state the following:

- The reference was matched quite well by almost all listeners for all samples, with slightly worst accuracy for Track 3 (Fig. 10c and 11c);

Figure 11: Test results, grouped by song. Box-plot of error in level evaluation versus the corresponding correct level.

- Fairly good results were obtained in typical music listening range (60 and 70 dB SPL);
- The variance increased towards the lowest levels (40 and 50 dB SPL), where sound was really quiet and the task of matching the perceived spectral balance became harder.

It is interesting to notice that, for Track 2 (Figs. 10b and 11b), the majority of the listeners tended to overcompensate when the music level went down. This makes sense, due to the sample having narrow spectral content and, as a



consequence, no “untouched” frequency components to be compared to, increasing the difficulty.

After taking the test, the subjects were asked for a feedback. They confirmed the difficulty of matching the spectral balance, when the level of reproduction went down. It was also difficult to notice the audible difference among small changes of the slider, since only the lowest frequencies were affected. They also stated that bass contribution was noticeable and pleasing.

## 7. CONCLUSION

A loudness compensation function derived from the equal-loudness-level contours and implemented via digital filters was proposed. This function introduces an adaptive contribution to the bass based on the listening level, in order to balance the perceived spectral variations given by the nonlinear response of the human hearing system.

Among different typologies, the first-order low-shelving filter with gain adjustment and fixed crossover frequency was shown to provide a high-fidelity approximation of the compensation function for a wide range of music listening levels. Its low computational complexity enables a real-time implementation.

A formal adaptive listening test was designed and conducted to validate the accuracy of the proposed compensation method, which was proved by the test results. Future work on this topic might include on-chip applications, customization for specific hardware or environments, and new listening tests conducted with a larger pool of non-trained listeners reflecting consumer market.

## Acknowledgments

This work was conducted between October 4, 2018 and January 22, 2019, when L. Fierro was visiting the Aalto Acoustics Lab. The support of the University of Brescia and Prof. P. Migliorati is gracefully acknowledged. This research was partially supported by NordForsk’s Nordic University Hub “Nordic Sound and Music Computing Network – NordicSMC”, project no. 86892. Special thanks are due to the Aalto Acoustics Lab staff members participating in the listening test.

## 8. REFERENCES

- [1] G. H. Fletcher and W. A. Munson, “Loudness, its definition, measurement, and calculation,” *J. Acoust. Soc. Am.*, vol. 5, no. 2, pp. 82–108, Oct. 1933.
- [2] D. W. Robinson and R. S. Dadson, “A redetermination of the equal-loudness relations for pure tones,” *Brit. J. Appl. Phys.*, vol. 7, pp. 156–181, May 1956.
- [3] A. L. Newcomb Jr. and R. N. Young, “Practical loudness: An active circuit approach,” *J. Audio Eng. Soc.*, vol. 24, no. 1, pp. 32–35, Jan./Feb. 1976.
- [4] T. Holman and F. Kampmann, “Loudness compensation: Use and abuse,” *J. Audio Eng. Soc.*, vol. 26, no. 7/8, pp. 526–536, Jul./Aug. 1978.
- [5] A. Seefeldt, “Loudness domain signal processing,” in *Proc. Audio Eng. Soc. 123<sup>rd</sup> Conv.*, New York, NY, USA, Oct. 2007.
- [6] P. D. Prasad, “A low complexity approach for loudness compensation,” in *Proc. Audio Eng. Soc. 129<sup>th</sup> Conv.*, San Francisco, CA, USA, Nov. 2010.
- [7] O. Hawker and W. Wang, “A method of equal loudness compensation for uncalibrated listening systems,” in *Proc. Audio Eng. Soc. 139<sup>th</sup> Conv.*, New York, NY, USA, Oct.-Nov. 2015.
- [8] B. Katz, *Mastering Audio: The Art and the Science*, 2<sup>nd</sup> ed. Butterworth-Heinemann, 2014.
- [9] NIOSH-CDC, *Occupational Noise Exposure—Revised criteria for a recommended standard*, 1998. [Online]. Available: <https://www.cdc.gov/niosh/docs/98-126/pdfs/98-126.pdf>
- [10] S. Widén, S. Båsjö, C. Möller, and K. Kähäri, “Headphone listening habits and hearing thresholds in Swedish adolescents,” *Noise & Health*, vol. 19, no. 88, pp. 125–132, May–June 2017.
- [11] S. B. Nielsen, “A loudness function for analog and digital sound systems based on equal loudness level contours,” in *Proc. Audio Eng. Soc. 140<sup>th</sup> Conv.*, Paris, France, June 2016, paper no. 9559.
- [12] ISO 226:2003, “Acoustics – Normal equal-loudness-level contours,” ISO/TC 43 Acoustics, International Standard, Aug. 2003.
- [13] V. Välimäki and J. D. Reiss, “All about audio equalization: Solutions and frontiers,” *Applied Sciences*, vol. 6, no. 5/129, May 2016.
- [14] J. O. Smith III, *Physical Audio Signal Processing: For Virtual Musical Instruments and Audio Effects*. W3K Publishing, 2010.
- [15] —, *Introduction to Digital Filters with Audio Applications*. W3K Publishing, 2007.
- [16] C. P. G. Tsirimokou and A. Elwakil, *Design of CMOS Analog Integrated Fractional-Order Circuits*. Springer, 2017.
- [17] U. Zölzer, *Digital Audio Signal Processing*, 2nd ed. Chichester, UK: Wiley, 2008.
- [18] S. U. Ahmad, “Design of digital filters using genetic algorithms,” Ph.D. dissertation, University of Victoria, Victoria, BC, Canada, Dec. 2008. [Online]. Available: <https://dspace.library.uvic.ca/handle/1828/1294>
- [19] IEC 60318-4, “Electroacoustics – Simulators of human head and ear – Part 4: Occluded-ear simulator for the measurement of earphones coupled to the ear by means of ear inserts,” IEC/TC 29 Electroacoustics, International Standard, Jan. 2010.
- [20] ITU-R BS.1770-4, “Algorithms to measure audio programme loudness and true-peak audio level,” ITU/Radiocommunications, standard, Oct. 2015.

# Toward Automatic Tuning of the Piano

**Joonas Tuovinen**

Acoustics Lab,  
Dept. Signal Processing & Acoustics,  
Aalto University, Espoo, Finland  
joonas.tuovinen@aalto.fi

**Jamin Hu**

Sibelius Academy,  
University of the Arts Helsinki,  
Helsinki, Finland  
jamin.hu@uniarts.fi

**Vesa Välimäki**

Acoustics Lab,  
Dept. Signal Processing & Acoustics,  
Aalto University, Espoo, Finland  
vesa.valimaki@aalto.fi

## ABSTRACT

The tuning of a piano is a complicated and time-consuming process, which is usually left for a professional tuner. To make the process faster and non-dependent on the skills of a professional tuner, a semi-automatic piano tuning system is developed. The aim of the system is to help a non-professional person to tune a grand piano with the help of a computer and a motorized tuning machine. The system composes of an aluminum frame, a stepper motor, an Arduino processor, a microphone, and a laptop computer. The stepper motor changes the tuning of the piano strings by turning the pins connected to them whereas the aluminum frame holds the motor in place. The Arduino controls the motor. The microphone and the computer are used as a part of a closed loop control system, which is used to tune the strings automatically. The control system tunes the strings by minimizing the difference between the current and optimal fundamental frequency. The current fundamental frequency is obtained with an inharmonicity coefficient estimation algorithm, and the optimal fundamental frequency is calculated with a novel tuning process, called the Connected Reference Interval (CRI) tuning. With the CRI process, a tuning close to that of a professional tuner is achieved with a deviation of 2.5 cents (RMS) between the keys  $A_0$  and  $G_5$  and 8.1 cents (RMS) between  $G\#_5$  and  $C_8$ , where the tuner's results are not very consistent.

## 1. INTRODUCTION

Tuning a piano is known to be a complicated process, which takes a considerable amount of time and effort. To many musicians tuning all the 200 plus strings of the instrument is a daunting task, especially as doing it incorrectly may leave the instrument in even worse tune. Because of this the tuning of a piano is usually left to professional tuners.

The scale of a piano is based on the twelve-tone equal temperament scale (12-ET), which specifies the fundamental frequency of each key. The difficulty of tuning a piano comes from the fact that, because mode frequencies

of piano strings deviate from the harmonic series, in a phenomenon called inharmonicity, tuning the fundamental frequencies of the strings to follow the 12-ET scale leads the instrument to sound out of tune [1]. Instead, professional tuners use the beating effect, produced by two frequencies close to each other, to tune the instrument, as the 12-ET scale specifies beating rates for each interval [2].

To make the process of tuning a piano faster and non-dependent on the skills of a professional tuner, a semi-automatic piano tuning system is developed in this work. There have been related previous developments, e.g. [3], but an automatic piano tuning system is still not commonly used. The proposed tuning system is aimed towards tuning a grand piano with the help of a non-professional tuner. The system includes a stepper motor, an aluminum frame, an Arduino Uno [4], a microphone and a computer.

The system uses closed loop control to change the fundamental frequency of a string from the current frequency to a target frequency. The current fundamental frequency is determined by an inharmonicity coefficient estimation algorithm. The target frequency is determined by a novel tuning process, called the Connected Reference Interval (CRI) tuning process, which calculates the optimal fundamental frequency for each string based on the beating rates between the current and previously tuned strings. The change from current to target fundamental frequency is implemented with a Proportional-Integral-Derivative (PID) controller, which is discussed later in more detail.

There have been previous algorithms that are designed to find the optimal tuning for a piano, but these algorithms require all or some of the strings to be recorded before tuning [5–7], unlike the CRI tuning process, which calculates everything while the tuning is done.

The paper is structured as follows. In Section 2 the structure of the piano tuning robot, in charge of turning the pins of the piano, is described. Next, in Section 3 the control system which automatically changes the tune of a string to a desired tuning is discussed. The system needs the current fundamental frequency, discussed in Section 4, and the target fundamental frequency, described in Section 5 to tune the string. The accuracy of the CRI tuning system is also evaluated in Section 5. Finally, Section 6 includes conclusion of the project as well as discussion about the future of the system.

Copyright: © 2019 Joonas Tuovinen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



Figure 1. Proposed piano tuning system with the Arduino processor near the midpoint of the picture and the stepper motor on its left-hand side, attached to the topmost aluminum bar.

## 2. STRUCTURE OF PIANO TUNING ROBOT

In the process of tuning a piano, tuners use a lever to tune the strings of the piano. The lever is used to turn a pin which has a string wrapped around it. Turning the pin changes the tension of a string and this change in tension determines the fundamental frequency of the string according to:

$$f_0 = \frac{1}{2L} \sqrt{\frac{T}{m/L}}, \quad (1)$$

where  $L$  is the length of the string,  $m$  is the mass of the string and  $T$  is the tension of the string. The fundamental frequency of a string (along with the inharmonicity coefficient which will be discussed later) determines the mode frequencies (partials) that tuners listen to when tuning the instrument.

The first step in making an automatic piano tuner was to create a structure which allows the automatic control of string tension. The proposed structure (stepper motor, aluminum frame and Arduino) is able to turn the pins of the piano with high precision (small angle) and has enough torque to turn even the tightest strings. The Arduino is able to turn the pins of the piano depending on input given by the computer with a program uploaded to it. The prototype structure can be seen in Figure 1.

## 3. CONTROL SYSTEM

The control system used to automatically determine the number of steps needed to change the fundamental frequency of a string from current to a desired value, is a closed loop control system. The general structure of a closed loop control system can be seen in Figure 2a. The system has a reference value as its input, and the aim of the control loop is to minimize the difference between the reference and the value measured from the output of the system with the sensor. This is accomplished with the controller, which changes the input to the process based on the difference between the reference and the measured output.

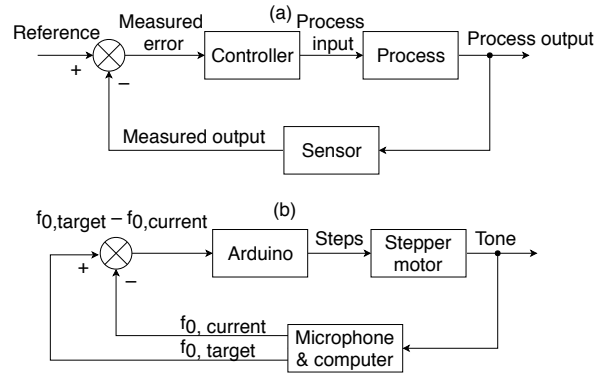


Figure 2. Block diagram of (a) a general closed loop control system (b) closed loop control system used in the piano tuning robot.

Figure 2b shows a diagram of the piano tuning system. In the piano tuning system the Arduino is used as the controller as the microprocessor has enough processing power to do this task and it controls the number of steps that the stepper motor takes. The process is the stepper motor turning a pin and thus affecting the fundamental frequency of the string attached and the process output is a tone produced by the string. To be able to measure the fundamental frequency of a string based on the tone, a microphone and computer are added to the system. In addition to working as the sensor, providing the measured output, the two components provide the reference as well, as information extracted from the current tone as well as previous tones is used to calculate the target frequency. The difference between the current and target fundamental frequency is used by the controller (Arduino) to calculate the appropriate number of steps the stepper motor should take.

A PID (proportional-integral-derivative) control scheme is used by the Arduino to control the stepper motor. PID controller calculates the error value  $e(t)$  between the reference ( $f_{0,target}$ ) and measured output ( $f_{0,current}$ ) and applies a correction to the process based on proportional (P), integral (I) and derivative (D) terms. This control value  $u(t)$  (number of steps) attempting to minimize the difference between the reference and measured output is then applied to the process.

## 4. FUNDAMENTAL FREQUENCY

The fundamental frequency of a string can be estimated looking at the spectrum of its tone. This is done by finding spectral peaks belonging to mode frequencies of the string. The relationship between these partials and the fundamental frequency is affected by an effect called inharmonicity. In this section, inharmonicity as well as algorithms for estimating the fundamental frequency of a string are reviewed.

### 4.1 Inharmonicity

The partials of an ideal string are integer multiples of its fundamental frequency (harmonics). However, real strings have stiffness, which acts as a restoring force, making the

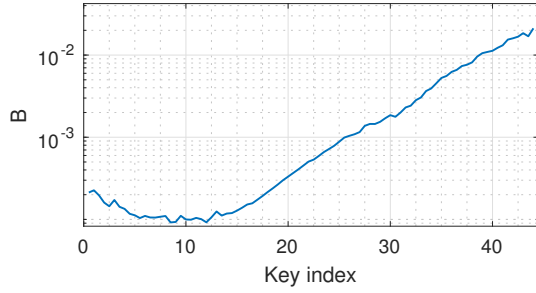


Figure 3. Inharmonicity coefficients of piano strings calculated with an inharmonicity coefficient estimation algorithm from tones recorded from a Yamaha grand piano. The strings for keys 1–7 are wrapped twice in wire, those of keys 8–26 are wrapped once, and the rest of the strings are unwrapped.

mode frequencies deviate from the harmonic series. This deviation is called inharmonicity and the effect is greater with higher modes, as they have more bends [8].

The mode frequencies of piano strings can be calculated from the following equation [9]:

$$f_k = k f_0 \sqrt{1 + B k^2}, \quad (2)$$

where  $f_k$  is the frequency of the  $k$ th mode (also known as the  $k$ th partial),  $f_0$  is the fundamental frequency, and  $B$  is the inharmonicity coefficient of the string. The value of inharmonicity coefficient of a solid string depends on the length, tension and radius of the string according to the following equation [8]:

$$B = \frac{\pi^3 r^4 E}{8 T L}, \quad (3)$$

where  $r$  is the radius of the string,  $E$  is Young's modulus,  $T$  is tension, and  $L$  is the length of the string. The strings in the bass end are wrapped in wire to lower their fundamental frequency by increasing their linear mass. This increases their inharmonicity slightly from Equation 3, but not as much as adding the linear mass by increasing the radius of the solid string.

Inharmonicity coefficient values of a Yamaha grand piano can be seen in Figure 3. It can be seen how the short strings in the treble end have inharmonicity coefficients close to  $1e^{-2}$  and the long bass strings have inharmonicity coefficients close to  $1e^{-4}$ . It can be also seen how the values of inharmonicity coefficients increases towards the bass end, as the strings are wrapped in one or two layers of wire.

## 4.2 Inharmonicity Coefficient Estimation

Inharmonicity coefficient estimation algorithms find spectral peaks belonging to partial frequencies and make estimations for the inharmonicity coefficient and fundamental frequency based those values. These algorithms need rough estimations for the values of  $B$  and  $f_0$  and make better ones based on the found partial frequencies. The difficulty of finding spectral peaks, belonging to partial

frequencies, comes from distinguishing partial frequencies from other spectral peaks.

There has been many algorithms tackling the issue of partials frequency estimation [10–14], but most of these algorithms suffer from high computational complexity. As the control system of the piano tuner needs the sensor (microphone and computer) to calculate the value of fundamental frequency on every iteration loop, the chosen algorithm has to be fast as well as accurate.

The Median-Adjustive Trajectories [14] (MAT) algorithm for estimating the inharmonicity coefficient best fulfills the accuracy and runtime requirements of the piano tuner. The algorithm calculates estimations for  $B$  and  $f_0$  based on the frequencies of known partials, and finds new partials based on these estimations. The algorithm is based on the idea that if the frequencies of two partials are known, the value of  $B$  can be calculated purely based on their values.

If equation 2 is solved in terms of fundamental frequency, with partial number  $m$ :

$$f_0 = \frac{f_m}{m \sqrt{1 + B m^2}}, \quad (4)$$

and then 4 is substituted into equation 2 for partial  $k$ :

$$f_k = k f_0 \sqrt{1 + B k^2} = k \frac{f_m}{m \sqrt{1 + B m^2}} \sqrt{1 + B k^2}. \quad (5)$$

The value of  $B$  can be solved from this equation:

$$B = \frac{(f_k \frac{m}{k})^2 - f_m^2}{k^2 f_m^2 - m^2 (f_k \frac{m}{k})^2}. \quad (6)$$

This means that if the frequencies of first two partials can be found from the spectrum of the tone, an estimation for the value of  $B$  can be made. As the first two partials do not deviate very much from the harmonic series, these can be found with good initial estimations of  $f_0$  and  $B$ . This new  $B$  estimation can then be used together with the found partial frequencies to make new estimations for  $f_0$  with Equation 4. After that, these  $f_0$  and  $B$  estimations can then be used to find new partial frequencies from the spectrum.

Figure 4 shows the block diagram of the MAT algorithm. The diagram shows how the initial estimates of the  $f_0$  and  $B$  are used to find the first two partials of the tone. The original MAT algorithm suggested that first two partials should be found by looking at a window around frequencies  $f_{0,init}$  and  $2f_{0,init}$ . A small adjustment to the algorithm is made by using the initial values of  $f_0$  and  $B$  in Equation 2 to calculate estimates for the first two partials. By doing so a slight improvement to the accuracy of the algorithm is achieved.

After the first two partials are found, the first  $B$  estimation can be made with Equation 6. This value is stored to an array of  $B$  estimates and a median of this array is taken to make two estimation for the value of  $f_0$ . The  $f_0$  estimations are then stored into an array of  $f_0$  estimations and the median values of the  $B$  and  $f_0$  arrays are used in Equation 2 to make an estimation for the value of the third partial  $f_3$ . A smaller window around the estimated value can be used to find the partial, as the estimate is more accurate than the



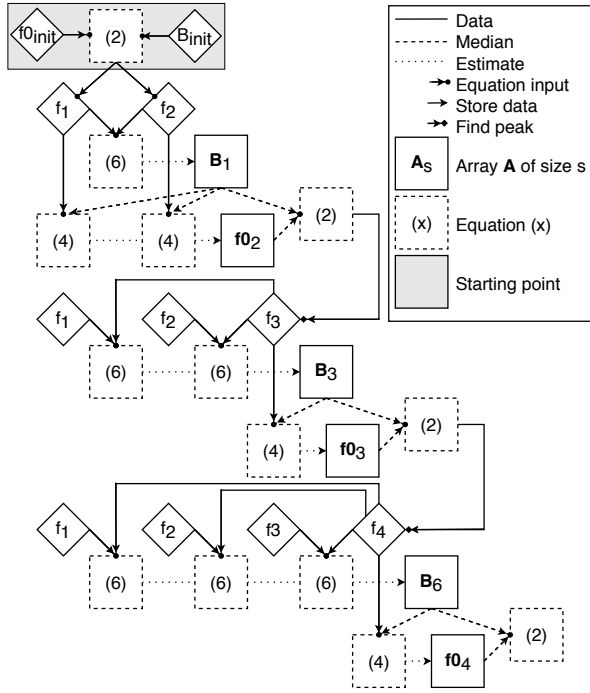


Figure 4. Block diagram of the MAT algorithm adopted from [14]

estimation for the first two partial. The process moves on making new estimations for  $B$ ,  $f_0$  and partial frequencies until the found partials have a magnitude below a specified threshold. At this point the median of the  $B$  and  $f_0$  estimation are then used for the final estimations.

With this method the estimation for the current value of  $f_0$  is gotten. These  $f_0$  and  $B$  estimates will also be used in the CRI tuning process, as they can be used in Equation 2 to represent the partial frequencies of the string, which are used to calculate beat rates between piano strings.

## 5. CRI TUNING PROCESS

The CRI tuning process determines a target frequency for every piano string. The process specifies the order of tuning so that as many intervals as possible can be used to calculate the target fundamental frequency. The keys of the piano are connected to one, two or three strings, and all the strings connected to the same key are called a string unison. The tuning process specifies that a single string from each string unison is to be tuned at first, using single strings from other unisons as a reference, and after that, the rest of the strings in the same unison are tuned using the tuned string as a reference. The other strings in the unison are tuned to have approximately a 1.5 cent difference to the reference as that maximizes the decay time of the combined strings [15]. From here on, the tuning process talks only about the single string of every string unison, which is tuned using single strings of other unisons as reference.

Strings are tuned one by one starting with a reference tone ( $A_4$ ) which is tuned to a reference frequency, and after that, the rest of this strings are tuned in the following order: ref-

erence octave ( $F_3$  to  $F_4$ ), tones above the reference octave ( $F\#_4$  to  $C_8$ ) and tones below the reference octave ( $E_3$  to  $A_0$ ).

The target fundamental frequency for each string is found by optimizing the beating rates between the string that is currently tuned and all the strings that have been already tuned and are a certain interval away from that string.

### 5.1 Beats

When a tone contains two frequencies that are close to each other, the frequencies cause periodic changes in the amplitude of the tone. These amplitude modulations are called beats and the frequency of these modulations can be calculated from equation [16]:

$$f_B = |f_2 - f_1| = \Delta f, \quad (7)$$

where  $f_1$  and  $f_2$  are the two frequencies close to each other. Equation 7 applies only until a certain point. As the two frequencies get further away from each other the frequency of beats gets faster at first, until unpleasant roughness between the two frequencies emerges. From this roughness two distinct tones can be heard after  $\Delta f$  exceeds the limit of frequency discrimination and after  $\Delta f$  surpasses the critical band, the roughness disappears and only two distinct frequencies can be heard [16].

### 5.2 Scale of the Piano

The tuning of a piano is based on the equal temperament scale, which makes all the steps in the scale equal. This means that the ratio between fundamental frequencies of subsequent tones in the scale should be the same. More specifically the scale is a twelve-tone equal temperament scale (12-ET) which in addition to having equal steps specifies the ratio between an octave to be 2:1 ( $f_0$  of the lower tone is two times the  $f_0$  of the higher one) and divides each octave into twelve steps. The ratio of 2:1 and 12 equal steps leads a single step in the scale to have a ratio of  $\sqrt[12]{2}:1$ , as  $12 \sqrt[12]{2} = 2$ .

The distance between two tones in a scale is called an interval. Musical scales are usually designed so that some partials of two harmonic tones having a certain interval line up to produce the minimum level of roughness between tones. This is achieved by designing fundamental frequencies of the intervals to have certain frequency ratios, as harmonic overtones are integer multiples of the fundamental. For example, if the fundamental frequencies of two tones have frequency ratio of 2:1, the  $2k$  partials of the lower tone match with the  $k$  partials of the higher one ( $k = 1, 2, 3, \dots$ ). The names of these intervals and ratios of their fundamental frequencies are listed in the first two columns of Table 1.

The way the 12-ET scale is designed leads all other interval ratios except for the octave to deviate. The amount of deviation per interval can be seen in the third column of Table 1. This deviation is measured in cents, which is a logarithmic unit, expressed as:

$$Deviation = 1200 \log_2(b/a), \quad (8)$$

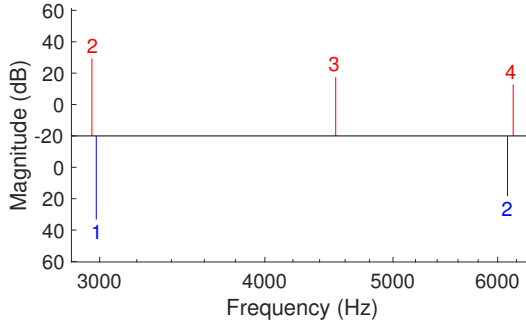


Figure 5. The first two matching partials in the octave between  $Bb_7$  and  $Bb_8$ .

where the deviation is positive if  $b$  is greater than  $a$ . The Distance column in Table 1 shows the number of steps (semitones) is between each interval in the 12-ET scale.

This deviation leads the intervals to have specific beating rates, which tuners use to tune the instrument. The reason why the fundamental frequencies of the 12-ET scale cannot be used to tune the instrument is because piano strings are inharmonic, and thus the spacing of fundamental frequencies specified by the scale do not produce wanted beat rates. Instead the spacing is slightly wider as the partials deviate upward from the harmonic series. This leads the tuning of the piano to be "stretched", meaning that when compared to the 12-ET scale, a tuning performed by a professional tuner is slightly higher in the treble and lower in the bass. Also, as the inharmonicity is different with each string, it is impossible for all the partials that are integer multiples of the frequency ratios to have the same beating rate. Because of this, piano tuners listen to all beats and tune the strings in such a way that none of the prominent beats deviate too much from the desired beating rate.

An example of this can be seen in Figure 5 which shows the partial frequencies and magnitudes of the octave between  $Bb_7$  and  $Bb_8$ . According to Table 1 the beating rate and thus the difference between the two sets of frequencies (2:1 and 4:2) should be zero, but this is not possible as matching either of the two sets would leave the other one to have even more deviation. Instead the tuner has made a compromise. The second partial of  $Bb_7$  is tuned

slightly below the frequency of the first partial of  $Bb_8$  and the fourth partial of  $Bb_7$  is tuned slightly above the second partial of  $Bb_8$ .

Electronic tuners that use partial frequencies of octaves to tune a piano match only a pair of partials. For example, a 2:1 method of tuning octaves matches only the second partial of the lower tone with the first partial of the higher tone and a 4:2 method of tuning octaves uses only the fourth and the second partial [6].

### 5.3 Calculating Target Frequencies

As the 12-ET scale specifies beating rates for each interval, and those beating rates should be calculated as the difference between the inharmonic partials of piano tones, the values of  $f_0$  and  $B$  obtained with the MAT algorithm can be used find the fundamental frequency that provide said beat rates. The beating rate in cents between two frequencies within a certain interval can be calculated from equation:

$$1200 \log_2 \left( \frac{f_{l,n+m}}{f_{k,n}} \right), \quad (9)$$

where  $c$  is the difference in cents,  $f_{k,n}$  is the  $k$ th partial of the  $n$ th tone and  $f_{l,n+m}$  is the  $l$ th partial of the tone which has a  $m$  semitone difference (interval) from  $n$ . The values of  $c$ ,  $k$ ,  $l$  and  $m$  for the first matching partials of each intervals can be seen in Table 2 (for other matching partials integer multiples of  $k$  and  $l$  are used and all other variables stay the same).

Equation 9 can be rewritten in terms of  $f_0$  and  $B$  by using Equation 2 to calculate partial frequencies:

$$1200 \log_2 \frac{l f_{0,n+m} \sqrt{1 + B_{n+m} l^2}}{k f_{0,n} \sqrt{1 + B_n k^2}} - c = 0. \quad (10)$$

When Equation 10 is used to calculate the sum of multiple intervals with the same tone  $n$ , the following equation is obtained:

$$\sum_{i=0}^{N-1} \left[ \log_2 \left( \frac{l_i f_{0,n+m_i} \sqrt{1 + B_{n+m_i} l_i^2}}{k_i f_{0,n} \sqrt{1 + B_n k_i^2}} \right) - C_i \right] = 0, \quad (11)$$

where  $N$  is the number of intervals used for the tuning and  $k_i$ ,  $l_i$ ,  $m_i$ , and  $c_i$  are the values  $k$ ,  $l$ ,  $m$ , and  $c$ , respectively,

Interval	Ratio	Deviation	Distance
Octave	2:1	0	12
Perfect fifth	3:2	-1.96	7
Perfect fourth	4:3	+1.96	5
Major sixth	5:3	+15.64	9
Major third	5:4	+13.69	4

Table 1. Several intervals. The ratio tells which partials of the lower tone is closes to the partial of the higher tone (partial of lower tone : partial of higher tone). The deviation tells how much deviation there is between these partials according to the 12-ET scale. The distance is the number of semitones between the two tones.

Interval	c	k	l	m
Octave (up)	0	2	1	+12
Perfect fifth (up)	-1.96	3	2	+7
Perfect fourth (up)	+1.96	4	3	+5
Major sixth (up)	+15.64	5	3	+9
Major third (up)	+13.69	5	4	+4
Octave (down)	0	1	2	-12
Perfect fifth (down)	+1.96	2	3	-7
Perfect fourth (down)	-1.96	3	4	-5
Major sixth (down)	-15.64	3	5	-9
Major third (down)	-13.69	4	5	-4

Table 2. Values of  $c$ ,  $k$ ,  $l$  and  $m$  for several intervals.

for a specific interval and  $C_i$  equals

$$C_i = 2^{c_i/1200}. \quad (12)$$

An estimation for the value of  $f_{0,n}$  can be made by solving it from Equation 11 with the assumption that the inharmonicity coefficient of the string does not change during the tuning. This estimate is fairly accurate as the change in tension changes  $f_0$  much more than  $B$ . Other coefficients in the equation are known, as  $n + m$  is the index of a previously tuned string with known values of  $f_0$  and  $B$ . It should be noted that the intervals used for this tuning are a design choice and that some intervals will get a tuning closer to that of a human tuner, as human tuners use only specific intervals to tune the instrument [2].

When  $f_{0,n}$  is solved from Equation 11, the following equation is obtained:

$$f_{0,n} = \left( \frac{\prod_{i=0}^{N-1} A_i}{\prod_{i=0}^{N-1} 2^{c_i/1200}} \right)^{1/N}, \quad (13)$$

where

$$A_i = \frac{l_i f_{0,n+m_i} \sqrt{1 + B_{n+m_i} l^2}}{k_i \sqrt{1 + B_n k_i^2}}. \quad (14)$$

The fundamental frequency of piano strings can be computed using Equation 13 by comparing multiple intervals. However, as the equation uses one set of partials per interval, the process does not take higher partials into consideration.

## 5.4 Weights

To take all the audible beats into consideration in a similar way as an human tuner does, weights can be added to Equation 13. The weight of a set of partials producing beating within an interval is calculated by taking the maximum loudness of the beating effect as well as masking into consideration. Masking is a phenomenon in which soft sounds cannot be heard because of loud ones occurring at the same time, or in other words, louder sounds mask softer sounds. When weights are added to equation 13, the following form is obtained:

$$f_{0,n} = \left( \frac{\prod_{i=0}^{N-1} A_i^{w_i}}{\prod_{i=0}^{N-1} 2^{c_i w_i/1200}} \right)^{1/\sum_{i=0}^{N-1} w_i}, \quad (15)$$

where  $w_i$  is the weight of a specific interval.

The weights are distributed in a way that the sum of the weights for each interval is one, so the weight of each interval is the same. The weights are calculated with the following steps:

1. Find the magnitude of partial frequencies: The magnitudes of partial frequencies can be found and stored by modifying the MAT algorithm to do so.
2. Apply A-weighting: The A-weighting is applied to approximate the frequency-dependent sensitivity of human hearing.

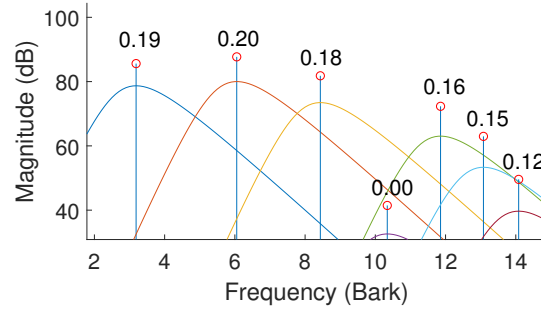


Figure 6. Weights (stems) of the beats between  $E_3$  and  $E_4$  (octave) and the corresponding masking thresholds.

3. Masking: An approximation of masking can be calculated by using a spreading function (SF). A popular SF proposed by Schroeder is used, as it is independent of the masking SPL, which is unknown [17]. The SF is shifted slightly lower depending on the tonality of the masker [18].

4. Weights: After all partials under the masking threshold have been taken out of consideration, the weights for each set of partials are calculated as

$$w_i = \frac{M_i}{\sum_{n=0}^{N-1} M_n}, \quad (16)$$

where  $w_i$  is the weight of  $i$ th matching partial,  $M_i$  is the maximum magnitude of the beats produced by the partials, and  $N$  is the total number of matching partials over the masking threshold.

Figure 6 shows the weights of the octave between  $E_3$  and  $E_4$ . It can be seen that the sum of these weights is one and that all beats under the threshold have a weight of zero.

## 5.5 Accuracy

The accuracy of the CRI tuning process was estimated by comparing it to a tuning performed by a professional piano tuner. The deviation (in cents) between the first partials of each tuning was used for the comparison. Single string recordings of all the 88 keys of a Yamaha grand piano were made the next day after tuning. The tuning accomplished by the CRI tuning process was emulated by resampling the recorded tones.

The accuracy was evaluated without weights, using the first matching partials for each interval, and with weights. The appropriate kind of distribution of weights for all partials of each interval could not be achieved yet, and too much weight was given to higher partials. This led to excessive amount of stretching, much more than that of the tuner. Because of this, only the first fifteen partials were considered for the algorithm with weights.

Both algorithms (with and without weights) use the same values and intervals for the reference tone and the reference octave.  $A_4$  is used as the reference tone and the first partial of this tone was tuned to match the first partial of  $A_4$  tuned by the tuner. This way the tunings could be compared. The

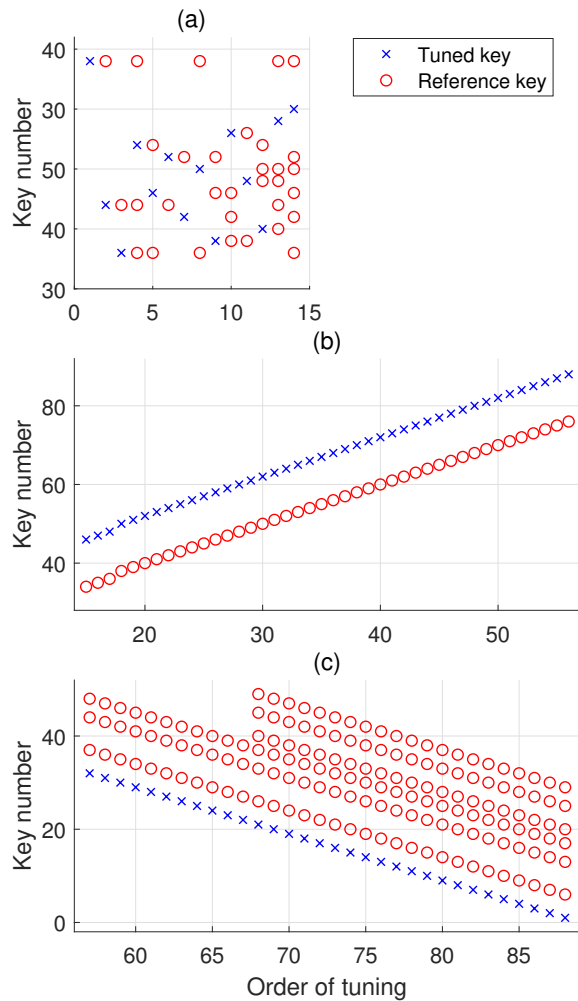


Figure 7. Order of tuning and reference intervals for the process without weights. Each cross ( $\times$ ) represents the key that is tuned while the circles (o) are the reference keys.

reference octave is tuned according to the “Defebaugh F-F” temperament, which is a tuning scheme commonly used by piano tuners [2].

For the algorithm without weights, matching the following intervals gave the best result:

- $F\#_4(46)$  to  $C_8(88)$ : Octaves
- $E_3(32)$  to  $F\#_2(22)$ : Octaves, fifths, tenths.
- $F_2(21)$  to  $A_0(1)$ : Octaves, fifths, tenths, seventeenth, double octaves, and double octaves and a third.

Figure 7 shows the order of tuning and the intervals used for the algorithm without weights. The crosses show the key that is being tuned whereas the circles above and below it are the keys that are used as a reference. For the algorithm with weights, using an octave and a double octave gave the best result.

Figure 8 shows the tuning curve produced by the algorithm without weights. Its deviation from the tuning conducted by the professional tuner is presented in Table 3. The tuning done by the algorithm with weights can be seen

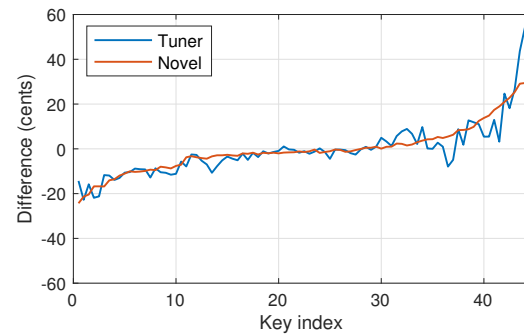


Figure 8. Deviation of the professional tuner and the CRI process (without weights) from 12-ET.

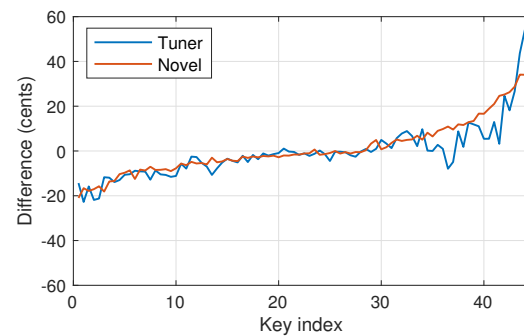


Figure 9. Deviation of the professional tuner and the CRI process with weights calculated from 15 first partials.

in Figure 9, and the corresponding deviations from the professional tuner’s result are presented in Table 4.

It can be seen that the algorithm without weights and optimized intervals gave a better accuracy with the overall deviation of 5.1 cents (RMS) than the one using weights, which had an overall deviation of 6.12 cents (RMS). It can also be seen that the deviation mostly happens in the treble end. This is most likely because these tones have a high degree of inharmonicity as well as a very short decay time, which make it harder for the tuner to hear and count beats.

## 6. CONCLUSIONS

In this paper a semi-automatic tuning system aimed toward tuning a grand piano with the help of a non-professional tuner was presented. The system uses a stepper motor attached to an aluminum frame to turn the tuning pins of the piano. The stepper motor is controlled by an Arduino pro-

Keys	RMS deviation (cents)
$A_0$ to $E_3$	3.2
Reference octave	1.3
$F\#_3$ to $C_8$	6.3
All	5.1

Table 3. Average deviation between the professional tuner and the CRI process without weights.



cessor, which is a part of a closed loop control system, to automatically adjust the tension of the strings. The control system also includes a microphone and a computer, which are used to measure the fundamental frequency and the inharmonicity coefficient of the piano strings as well as the magnitudes of partial frequencies from the tone of the string. The frequency values are obtained using an inharmonicity coefficient estimation algorithm called MAT and are used to calculate the difference between the current and the target fundamental frequency.

The target fundamental frequency is determined with a CRI tuning process using beating rates between the partials of several intervals. The process specifies the order of tuning for the strings to get the maximum number of intervals for its estimation. The process can calculate the fundamental frequency for either a specified set of partials, or for all partials, using weights. The process with and without specified partials were compared to a tuning conducted by a professional tuner. The process with specified partials was based on the first matching partials of each interval. With the optimal intervals, both processes gave great results with an RMS of 5.1 cents of deviation with specified partials and 5.7 cents without them.

#### Acknowledgments

This work was supported in part by NordForsk's Nordic University Hub "NordicSMC", project no. 86892. Jamin Hu would like to thank Mr. Kari Kääriäinen, Master Model Maker at Aalto Design Factory, for his generous advice on and fabrication of the custom parts of the aluminum frame.

#### 7. REFERENCES

- [1] D. W. Martin and W. Ward, "Subjective evaluation of musical scale temperament in pianos," *J. Acoust. Soc. Amer.*, vol. 33, no. 5, pp. 582–585, May 1961.
- [2] A. A. Reblitz, *Piano Servicing, Tuning, & Rebuilding: For the Professional, the Student, the Hobbyist*. Vestal Press, 1976.
- [3] D. A. Gilmore, "Apparatus and method for self-tuning a piano," Patent No. US 6,559,369B1, May 2003.
- [4] "Arduino uno rev3," <https://store.arduino.cc/arduino-uno-rev3>, accessed: 2019-04-04.
- [5] J. Lattard, "Influence of inharmonicity on the tuning of a piano—Measurements and mathematical simulation," *J. Acoust. Soc. Amer.*, vol. 94, no. 1, pp. 46–53, July 1993.
- [6] H. Hinrichsen, "Entropy-based tuning of musical instruments," *Revista brasileira de Ensino de Física*, vol. 34, no. 2, pp. 1–8, 2012.
- [7] F. Rigaud, B. David, and L. Daudet, "A parametric model of piano tuning," in *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 393–399.
- [8] T. D. Rossing, F. R. Moore, and P. A. Wheeler, *The Science of Sound*. Addison Wesley, 1990.
- [9] H. Fletcher, E. D. Blackham, and R. Stratton, "Quality of piano tones," *J. Acoust. Soc. Amer.*, vol. 34, no. 6, pp. 749–761, June 1962.
- [10] A. Galembo and A. Askenfelt, "Measuring inharmonicity through pitch extraction," *Journal STL-QPSR*, vol. 35, no. 1, pp. 135–144, 1994.
- [11] —, "Signal representation and estimation of spectral parameters by inharmonic comb filters with application to the piano," *IEEE Trans. Speech Audio Process.*, vol. 7, no. 2, pp. 197–203, Mar. 1999.
- [12] J. Rauhala and V. Välimäki, "F0 estimation of inharmonic piano tones using partial frequencies deviation method," in *Proc. Int. Computer Music Conf. (ICMC-07)*, Copenhagen, Denmark, Aug. 2007, pp. 453–456.
- [13] J. Rauhala, H.-M. Lehtonen, and V. Välimäki, "Fast automatic inharmonicity estimation algorithm," *J. Acoust. Soc. Amer.*, vol. 121, no. 5, pp. EL184–EL189, May 2007.
- [14] M. Hodgkinson, J. Wang, J. Timoney, and V. Lazzarini, "Handling inharmonic series with median-adjustive trajectories," in *Proc. 12th Int. Conf. Digital Audio Effects (DAFx-09)*, Como, Italy, Sept. 2009, pp. 1–7.
- [15] T. C. Hundley, H. Benioff, and D. W. Martin, "Factors contributing to the multiple rate of piano tone decay," *J. Acoust. Soc. Amer.*, vol. 64, no. 5, pp. 1303–1309, Nov. 1978.
- [16] J. G. Roederer, *The Physics and Psychophysics of Music: An Introduction*. Springer, 2001.
- [17] M. R. Schroeder, B. S. Atal, and J. L. Hall, "Optimizing digital speech coders by exploiting masking properties of the human ear," *J. Acoust. Soc. Amer.*, vol. 66, no. 6, pp. 1647–1652, Dec. 1979.
- [18] Y. Lin and W. H. Abdulla, "Audio watermarking techniques," in *Audio Watermark*. Springer, 2015, pp. 51–94.

Keys	RMS deviation (cents)
$A_0$ to $E_3$	3.3
Reference octave	1.4
$F\#_3$ to $C_8$	7.6
All	5.7

Table 4. Average deviation between a professional tuner and the CRI process with weights from first 15 partials.

# REAL-TIME CONTROL OF LARGE-SCALE MODULAR PHYSICAL MODELS USING THE SENSEL MORPH

**Silvin Willemsen, Nikolaj Andersson, Stefania Serafin**

Multisensory Experience Lab, CREATE,

Aalborg University Copenhagen

Copenhagen, Denmark

{sil, nsa, sts}@create.aau.dk

**Stefan Bilbao**

Acoustics and Audio Group

University of Edinburgh

Edinburgh, UK

s.bilbao@ed.ac.uk

## ABSTRACT

In this paper, implementation, instrument design and control issues surrounding a modular physical modelling synthesis environment are described. The environment is constructed as a network of stiff strings and a resonant plate, accompanied by user-defined connections and excitation models. The bow, in particular, is a novel feature in this setting. The system as a whole is simulated using finite difference (FD) methods. The mathematical formulation of these models is presented, alongside several new instrument designs, together with a real-time implementation in JUCE using FD methods. Control is through the Sensel Morph.

## 1. INTRODUCTION

Physical models for sound synthesis have been researched for several decades to mathematically simulate the sonic behaviour of musical instruments and everyday sounds. Various techniques and methodologies have developed, ranging from mass-spring models [1–3] to modal synthesis [4] and waveguide based models [5]. The latter two techniques may be viewed as numerical simulation techniques applied to the systems of partial differential equations (PDEs). These equations define the dynamics of a musical instrument, either real or imagined.

Mainstream time-domain simulation techniques, such as finite difference (FD) methods, were first applied to the case of string vibration by Ruiz [6] and Hiller and Ruiz [7, 8], and then later by other authors [9] including, most notably Chaigne [10] and Chaigne and Askenfelt [11]. The general use of finite-difference schemes (FDSs) in sound synthesis is described in [12]. Modularized physical modelling sound synthesis, whereby the user may construct a virtual instrument using basic canonical components dates back to the work of Cadoz and collaborators [1–3]. It has been also used as a design principle in the context of FD methods [13–15], where the canonical elements are strings and plates, with a non-linear connection mechanism. Though computational cost of such methods is high,

standard computing power is now approaching a level suitable for real-time performance for simpler systems.

We are interested in bridging the gap between large-scale modular physical modelling synthesis and sonic interaction design [16], to be able to play with such simulations in real-time. Specifically, we are interested in using the expressivity of the Sensel Morph [17] to control our simulations, using both percussive and bowing excitations. Our ultimate goal is to create models that are both mathematically accurate and efficient. This goal is nowadays possible thanks to improvements in hardware and software technologies for sound synthesis, yet it has rarely been achieved. The ultimate goal is to provide a modular efficient synthesizer based on accurate simulations, where real-time expressivity can also be achieved. This synthesizer has already been informally evaluated by composers and sound designers, who appreciated the current sonic palette.

This paper is structured as follows: Section 2 describes the physical models used in the implementation and Section 3 shows a general description of the FD methods used to digitally implement these models. Furthermore, Section 4 elaborates on the real-time implementation, Section 5 shows several different configurations of the physical models inspired by real musical instruments, Section 6 will present the results on CPU usage and evaluation and discuss this and finally, in Section 7, some concluding remarks appear.

## 2. MODELS

In this section, the PDEs for the damped stiff string and plate will be presented. The notation used will be the one found in [12] where the subscript for state variable  $u$  denotes a single derivative with respect to time  $t$  or space  $x$  respectively. Furthermore, to simplify the presented physical models, non-dimensionalization (or scaling) will be used [12].

### 2.1 Stiff string

A basic model of the linear transverse motion of a string of circular cross section may be described in terms of several parameters: the total length  $L$  (in m), the material density  $\rho$  (in  $\text{kg}\cdot\text{m}^{-3}$ ), string radius  $r$  (in m), Young's modulus  $E$  (in Pa), tension  $T$  (in N), and two loss parameters  $\sigma_0$  and  $\sigma_1$ .

Copyright: © 2019 Silvin Willemsen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

The PDE for a damped stiff string may be written as [12]

$$u_{tt} = \gamma^2 u_{xx} - \kappa^2 u_{xxxx} - 2\sigma_0 u_t + 2\sigma_1 u_{txx}. \quad (1)$$

In this representation, spatial scaling has been employed using a length  $L$ , so the solution  $u = u(x, t)$  is defined for  $t \geq 0$  and for dimensionless coordinate  $x \in [0, 1]$ . Furthermore, parameters  $\gamma = \sqrt{T/\rho\pi r^2 L^2}$  and  $\kappa = \sqrt{Er^2/4\rho L^4}$  and have units  $s^{-1}$ .

In this work, the string is assumed clamped at both ends, so that

$$u = u_x = 0 \quad \text{where} \quad x = \{0, 1\}. \quad (2)$$

A model of a bowed string [12] may be incorporated into (1) as

$$u_{tt} = \dots - \delta(x - x_B) F_B \phi(v_{\text{rel}}), \quad \text{with} \quad (3a)$$

$$v_{\text{rel}} = u_t|_{(x=x_B)} - v_B, \quad (3b)$$

where  $F_B = f_B/M_s$  is the excitation function (in  $m/s^2$ ) with externally-supplied bowing force  $f_B = f_B(t)$  (in N) and total string mass  $M_s = \rho\pi r^2 L$  (in kg). The relative velocity  $v_{\text{rel}}$  is defined as the difference between the velocity of the string at bowing point  $x_B$  and the externally-supplied bowing velocity  $v_B = v_B(t)$  (in m/s) and  $\phi$  is a dimensionless friction characteristic, chosen here as [12]

$$\phi(v_{\text{rel}}) = \sqrt{2a} v_{\text{rel}} e^{-av_{\text{rel}}^2 + 1/2}. \quad (4)$$

Furthermore,  $\delta(x - x_B)$  is a spatial Dirac delta function selecting the bowing location  $x = x_B$ . The single bowing point can be extended to a bowing area [12]. More detailed models of string dynamics, again in a bowed string context, have been proposed by Desvages [18].

Another, and more simple way to excite the string is by extending Equation (1) to

$$u_{tt} = \dots + E_e F_e \quad (5)$$

using an externally-supplied distribution function  $E_e = E_e(x)$  and excitation function  $F_e = F_e(t)$ . In this case, the excitation region is allowed to be of finite width.

## 2.2 Plate

Under linear conditions, a rectangular plate of dimensions  $L_x$  and  $L_y$  may be parameterized in terms of density  $\rho$  (in  $kg \cdot m^{-3}$ ), thickness  $H$  (in m), Young's modulus  $E$  (in Pa) and a dimensionless Poisson's ratio  $\nu$ , as well as two loss parameters  $\sigma_0$  and  $\sigma_1$ .

In terms of dimensionless spatial coordinates  $x$  and  $y$  scaled by  $\sqrt{L_x L_y}$ , the equation of motion of a damped plate is a variant of the Kirchhoff model [19]

$$u_{tt} = -\kappa^2 \Delta \Delta u - 2\sigma_0 u_t + 2\sigma_1 \Delta u_t. \quad (6)$$

Here,  $u(x, y, t)$  is the transverse displacement of the plate as a function of dimensionless coordinates  $x \in [0, \sqrt{a}]$ ,  $y \in [0, 1/\sqrt{a}]$ , where  $a = L_x/L_y$  is the plate aspect ratio, as well as time  $t$ . Furthermore,  $\Delta$  represents the 2D Laplacian [12]:

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}. \quad (7)$$

The stiffness parameter  $\kappa$ , with dimensions of  $s^{-1}$ , is defined by  $\kappa = \sqrt{D/\rho H L_x^2 L_y^2}$  where  $D = EH^3/12(1-\nu^2)$ . As in the case of the stiff string, we chose to use clamped boundary conditions:

$$u = \mathbf{n} \cdot \nabla u = 0 \quad (8)$$

over any plate edge with outward normal direction  $\mathbf{n}$  and where  $\nabla u$  is the gradient of  $u$ .

## 2.3 Connections

Adding connections between different physical models, further referred to as elements, adds another term to Equation (3a), (5) or (6). Assuming that element  $\alpha$  is a stiff string and  $\beta$  is a plate, the following terms are added to the aforementioned equations:

$$u_{tt} = \dots + E_{c,\alpha} F_\alpha, \quad (9a)$$

$$u_{tt} = \dots + E_{c,\beta} F_\beta, \quad (9b)$$

with force-functions  $F_\alpha = F_\alpha(t)$  and  $F_\beta = F_\beta(t)$  (in  $m/s^2$ ) and distribution functions  $E_{c,\alpha}$  and  $E_{c,\beta}$  which have chosen to be highly localised in our application and reduce to  $\delta(x - x_{c,\alpha})$  and  $\delta(x - x_{c,\beta}, y - y_{c,\beta})$  respectively, but can be extended to be connection areas [13]. We use the implementation as presented in [13] where the connection between two elements is a non-linear spring. The forces it imposes on the elements it connects are defined as

$$F_\alpha = -\omega_0^2 \eta - \omega_1^4 \eta^3 - 2\sigma_\times \dot{\eta}, \quad (10a)$$

$$F_\beta = -\mathcal{M} F_\alpha, \quad (10b)$$

where  $\omega_0$  and  $\omega_1$  are the linear (in  $s^{-1}$ ) and non-linear (in  $(m \cdot s)^{-1/2}$ ) frequencies of oscillation respectively,  $\sigma_\times$  is a damping factor (in  $s^{-1}$ ),  $\mathcal{M}$  is the mass ratio between the two elements and  $\eta$  is the relative displacement between the connected elements at the point of connection (in m). Lastly, the dot above  $\eta$  denotes a derivative with respect to time.

## 3. FINITE-DIFFERENCE SCHEMES

To be able to digitally implement the continuous equations shown in the previous section, they need to be approximated. In this section, a high-level review of a finite difference approximation to a connected system of strings and plates is presented. For more technical details, see [13].

In the case of the stiff string, state variable  $u(x, t)$  can be discretised at times  $t = nk$ , where  $n \in \mathbb{N}$  and  $k = 1/f_s$  is the time step (at sample-rate  $f_s$ ) and locations  $x = lh$ , with  $l \in [0, \dots, N]$  where the total number of points is  $N + 1$  and grid spacing  $h$ . We can now write the discretised state variable as  $u_l^n$ , representing an approximation to  $u(x, t)$ .

In the case of the plate,  $u(x, y, t)$  is discretised to  $u_{(l,m)}^n$  using  $x = lh$  where  $l \in [0, \dots, N_x]$  with  $N_x + 1$  being the total horizontal number of points and  $y = mh$  where  $m \in [0, \dots, N_y]$  with  $N_y + 1$  being the total vertical number of points.

In a general sense, when discretising PDEs as presented in Equations (1) and (6), we will need to solve for  $\mathbf{u}^{n+1}$ ,

i.e.,  $\mathbf{u}$  at the next time step, where  $\mathbf{u}$  is a vector of size  $N - 1$  containing values of  $u_l \forall l$  for a string and  $(N_x - 1)(N_y - 1)$  containing values of  $u_{(l,m)} \forall (l, m)$  for a plate. Note that the vector sizes are smaller than the total number of grid points as we do not include the values at the boundaries (which are always 0). For a PDE expressed as a function of  $u_{tt}$ , its FDS will be of the form

$$\mathbf{u}^{n+1} = 2\mathbf{u}^n - \mathbf{u}^{n-1} + K\mathcal{F}^n, \quad (11)$$

where

$$K = \frac{k^2}{1 + \sigma_0 k}, \quad (12)$$

and  $\mathcal{F}^n$  is a combination of the discretised PDE (excluding terms containing  $u^{n+1}$ ) together with connection and excitation terms.

### 3.1 Stiff String

In the case of the stiff string,  $\mathcal{F}^n$  in Equation (11) is a combination of the discretised PDE (1)  $\mathbf{f}_\alpha^n$ , connection term (9a) and bowing (3a)

$$\mathcal{F}^n = \mathbf{f}_\alpha^n + \mathbf{E}_{c,\alpha} F_\alpha^n - \mathbf{J}(x_B^n) F_B^n \phi(v_{\text{rel}}), \quad (13a)$$

or excitation (5) term

$$\mathcal{F}^n = \mathbf{f}_\alpha^n + \mathbf{E}_{c,\alpha} F_\alpha^n + \mathbf{E}_e F_e^n, \quad (13b)$$

where  $\mathbf{E}_{c,\alpha}$  contains the discretised distribution function for the connection ( $1/h$  at connection index  $l_{c,\alpha}$ , rest 0's [12]),  $\mathbf{E}_e$  contains the discretised distribution function for the excitation (which will be presented in Equation (25) in the next section) and  $\mathbf{J}(x_B^n)$  is a spreading operator containing the discretised bowing distribution ( $1/h$  at time-varying bowing position  $x_B$ ). If  $x_B$  is between grid points, cubic interpolation is used to spread the bow-force over neighbouring grid points [12]. All vectors are columns of size  $N - 1$ .

It can be useful to talk about the *region of operation* of a FDS in terms of a 'stencil'. A stencil describes the number of grid points needed to calculate a single point at the next time step. The stiff string FDS has a stencil of 5 grid points. In other words, two grid points at either side of  $l$  – and  $l$  itself – are necessary to calculate  $u_l^{n+1}$ . See Figure 1 for a visualisation of this.

In order for the scheme to be stable, the grid spacing needs to abide the following condition [12]

$$h \geq \sqrt{\frac{\gamma^2 k^2 + 4\sigma_1 k + \sqrt{(\gamma^2 k^2 + 4\sigma_1 k)^2 + 16\kappa^2 k^2}}{2}}. \quad (14)$$

The closer  $h$  is to this limit, the higher the quality of the implementation. The number of points  $N$  can then be calculated using

$$N = \text{floor}\left(\frac{1}{h}\right). \quad (15)$$

### 3.2 Plate

In the case of the plate,  $\mathbf{u}$  is a column vector of concatenated vertical 'strips' of the plate state as in [13] of size

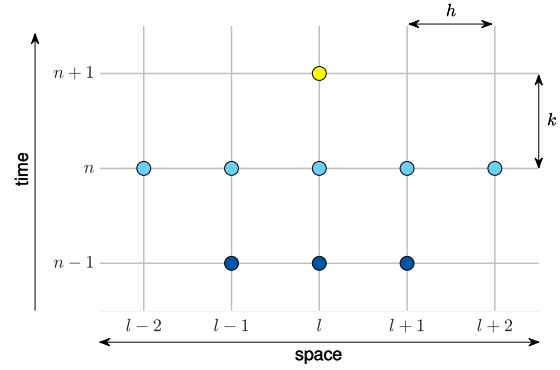


Figure 1. Stencil for a stiff string FDS with grid spacing  $h$  and time step  $k$ . The point  $l$  at the next time step (yellow) is calculated using 5 points at the current time step (blue) and 3 at the previous time step (dark blue).

$(N_x - 1)(N_y - 1)$  and  $\mathcal{F}^n$  in Equation (11) is a combination of the discretised PDE (6)  $\mathbf{f}_\beta^n$  and connection term (9b)

$$\mathcal{F}^n = \mathbf{f}_\beta^n + \mathbf{E}_{c,\beta} F_\beta^n. \quad (16)$$

Here,  $\mathbf{E}_{c,\beta}$  contains the discretised distribution function for the connection ( $1/h^2$  at connection index  $(l_{c,\beta}, m_{c,\beta})$ , rest 0's [13]) and is a column vector of size  $(N_x - 1)(N_y - 1)$ . For the plate, the stencil will consist of 13 grid points as can be seen in Figure 2.

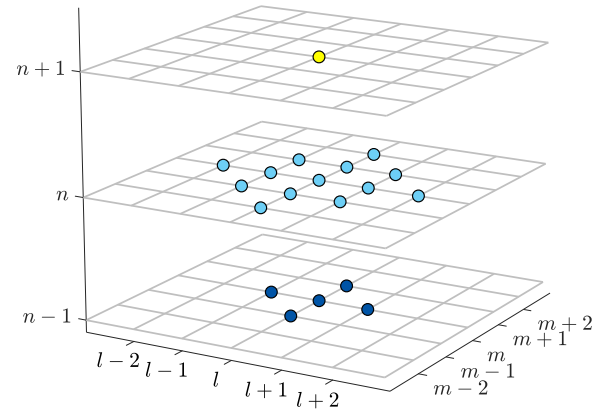


Figure 2. Stencil for a plate FDS. The point  $(l, m)$  at the next time step (yellow) is calculated using 13 points at the current time step (blue) and 5 at the previous time step (dark blue).

The grid spacing needs to abide the following condition [13]

$$h \geq 2\sqrt{k\left(\sigma_1^2 + \sqrt{\kappa^2 + \sigma_1^2}\right)}, \quad (17)$$

(again, the closer  $h$  is to this limit the better) from which



$N_x$  and  $N_y$  can be derived using

$$N_x = \text{floor}\left(\frac{\sqrt{a}}{h}\right) \quad \text{and} \quad N_y = \text{floor}\left(\frac{1}{h\sqrt{a}}\right). \quad (18)$$

### 3.3 Connections

In the following, we discretise the equations in (10) as shown in [13]. However, as these equations are not expressed as a function of  $u_{tt}$ , their FDS counterpart will be different. Moreover, instead of solving for  $\mathbf{u}^{n+1}$ , we need to solve for  $\eta^{n+1}$ , i.e., the relative displacement at the next time step, which will be in the form of

$$\eta^{n+1} = p^n F_\alpha^n + r^n \eta^{n-1}, \quad (19)$$

where  $p^n = p(\eta^n)$  and  $r^n = r(\eta^n)$  are functions of the relative displacement  $\eta$  if  $\omega_1 \neq 0$  and constants if  $\omega_1 = 0$ . Again, assuming that element  $\alpha$  is a stiff string and  $\beta$  is a plate,  $\eta$  can be calculated using

$$\eta^n = h_\alpha u_{\alpha, l_{c,\alpha}}^n - h_\beta^2 u_{\beta, (l_{c,\beta}, m_{c,\beta})}^n. \quad (20)$$

In other words, this is the difference between the state of element  $\alpha$  at  $l_{c,\alpha}$  and the state of element  $\beta$  at  $(l_{c,\beta}, m_{c,\beta})$  scaled by their respective (for plates, squared) grid spacings  $h_\alpha$  and  $h_\beta$ . The next step is to obtain  $F_\alpha^n$ , which can be used to easily calculate  $F_\beta^n$ . We first obtain values for  $\mathbf{u}^{n+1}$  by solving (11) using (13a), (13b) or (16) (without the connection term!) for a string or plate respectively. As, at this point, no connection forces have been added yet, this state will be referred to as an intermediate state  $\mathbf{u}^1$ . This intermediate state can be used to obtain  $\eta^{n+1}$  using (20)

$$\eta^{n+1} = h_\alpha (u_{\alpha, l_{c,\alpha}}^1 + K_\alpha F_\alpha^n) - \left[ h_\beta^2 (u_{\beta, (l_{c,\beta}, m_{c,\beta})}^1 + K_\beta F_\beta^n) \right], \quad (21)$$

where  $K_\alpha$  and  $K_\beta$  are as described in (12) using the damping coefficient  $\sigma_0$  of their respective element. This can then be set equal to (19). Using Equation (10b), solving for  $F_\alpha$  yields

$$F_\alpha^n = \frac{r^n \eta^{n-1} - (h_\alpha u_{\alpha, l_{c,\alpha}}^1 - h_\beta^2 u_{\beta, (l_{c,\beta}, m_{c,\beta})}^1)}{h_\alpha K_\alpha + \mathcal{M} h_\beta^2 K_\beta - p^n}. \quad (22)$$

## 4. IMPLEMENTATION

In this section, we elaborate more on the chosen values for the parameters described in the previous two sections and present the system architecture of the real-time application. The values for most parameters have been arbitrarily chosen and can – as long as they satisfy the conditions in Equations (14) and (17) – be changed. We used C++ along with the JUCE framework [20] for implementing the physical models and connections in real-time. The main hardware used was a MacBook Pro with a 2.2 GHz Intel Core i7 processor.

### 4.1 Stiff String

As many string properties stay constant, we chose to set the following parameters directly, rather than calculating

them from their physical properties:  $\kappa = 2$ ,  $\sigma_0 = 1$ ,  $\sigma_1 = 0.005$ . An interesting parameter to make dynamic is the fundamental frequency  $f_0$  (in  $\text{s}^{-1}$ ) of the string. According to [12], the fundamental frequency can be approximately calculated using

$$f_0 \approx \frac{\gamma}{2}. \quad (23)$$

However, as the grid spacing  $h$  is dependent on the wave speed  $\gamma$  according to the condition found in (14), we must put a lower limit on the number of points  $N$  if we plan to dynamically increase  $\gamma$ .

Another way to change frequency is to add damping to the model at specific points acting as a (simplified) fretting finger. The advantage of this is that the condition (14) will never be violated. On top of this, a tapping sound will be introduced when fretting the string making it more realistic than changing the wave speed. If the string is fretted at single location  $x_f \in [0, 1]$  and  $l_f = \text{floor}(x_f/h)$  we use

$$u_l^n = \begin{cases} 0, & l = l_f - 1 \vee l = l_f \\ (1 - \alpha_f^\epsilon) u_l^n, & l = l_f + 1 \\ u_l^n, & \text{otherwise} \end{cases} \quad (24)$$

where  $\alpha_f = x_f/h - l_f$  describes the fractional location of  $x_f$  between two grid points. Note that the grid point at the finger location and the grid point before are set to 0 to (recalling the stencil) prevent the states at either side of the finger to influence each other. The disadvantage of using this technique over regular linear interpolation, is that the effect of damping between grid points does not linearly scale to pitch. We thus added  $\epsilon = 7$  as a heuristic value to more properly map finger position to pitch.

In some cases,  $N$  is fixed to a certain value (as opposed to calculating it from Equations (14) and (15)) for multiple strings of different pitches. Even though some bandwidth will be lost (in the higher frequency range), this will allow the strings to be perfectly tuned to each other.

#### 4.1.1 Bowed String

Parameters for the bowed strings abide the following conditions:  $|v_B| \leq 1 \text{ m/s}$  and  $0 \leq F_B \leq 100 \text{ N}$ . It was chosen to discretise Equation (3b) implicitly making it necessary to use an iterative root-finding method such as Newton-Raphson [21].

#### 4.1.2 Excited string

If simply excited, we set the distribution function to a raised cosine with width  $w_e$  (in grid points)

$$E_e(l) = \begin{cases} \frac{1 - \cos\left(\frac{2\pi(l - (l_e - w_e/2))}{w_e}\right)}{2}, & l_e - \frac{w_e}{2} < l < l_e + \frac{w_e}{2} \\ 0, & \text{otherwise} \end{cases} \quad (25)$$

scaled by the excitation function over time with excitation duration  $d_e$  (in samples)

$$F_e(n) = \begin{cases} \frac{1 - \cos\left(\frac{\pi(n - n_e)}{d_e}\right)}{2}, & n_e \leq n < n_e + d_e \\ 0, & \text{otherwise} \end{cases} \quad (26)$$

A visualisation of this can be found in Figure 3.

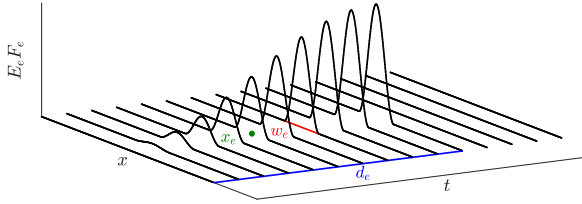


Figure 3. A visualisation of the excitation used in our implementation presented in Equation (5). The location of excitation  $x_e$  is shown in green, excitation width  $w_e$  in red and excitation duration  $d_e$  in blue (also see Equations (25) and (26)).

## 4.2 Plate

For the plate, the damping coefficients have been decided to be  $\sigma_0 = 0.1$  and  $\sigma_1 = 0.005$  and the aspect ratio is set to  $a = 2$ . The plate stiffness  $\kappa$  has been left as a user parameter to be changed dynamically and will be between the following bounds:  $0.1 \leq \kappa \leq 50 \text{ s}^{-1}$ . In Equation (17), the grid spacing is calculated using the maximum value of  $\kappa$  to prevent stability issues. Using a sample rate of 44,100 Hz results in a plate with dimensions  $N_x = 20$  and  $N_y = 10$  (in grid points).

## 4.3 Connections

Increasing  $\omega_1 \gtrsim 100,000 \text{ (m}\cdot\text{s)}^{-1/2}$  while keeping  $0 < \omega_0 \lesssim 100 \text{ s}^{-1}$  will cause audible non-linear behaviour, such as pitch-glides and rattling sounds. These effects will be more dominant when the plate stiffness is higher. In our implementation we set  $\omega_0 = 100 \text{ s}^{-1}$  and  $\omega_1 = 100,000 \text{ (m}\cdot\text{s)}^{-1/2}$ . The spring-damping  $\sigma_x = 1 \text{ s}^{-1}$  is kept to a minimum ( $0 \leq \sigma_x \leq 10 \text{ s}^{-1}$ ).

## 4.4 System Architecture

The system architecture can be seen in Figure 4. The top box denotes the Sensel Morph (described in more detail in the next section) controlling the application, and the white boxes show the different classes or components of the application. The black arrows indicate instructions that one class can give to another and the hollow arrows show data flows between classes. All arrows are accompanied by a coloured box indicating which thread the instruction / dataflow is associated with and at what rate this thread runs.

The lowest priority thread, the graphics-thread, is shown by green boxes and runs at 15 Hz. This draws the states of the strings, connections and the plate on the screen.

Checking and retrieving the Sensel state happens at a rate of 150 Hz and is denoted by blue boxes. The parameters that the user controls by means of the Sensel, such as bowing position, force and velocity, will be updated in the models at this rate as well.

The highest priority thread is the audio-thread and runs at commonly-used sample rate 44,100 Hz. The main application gives an ‘update’ (u) instruction to the instrument,

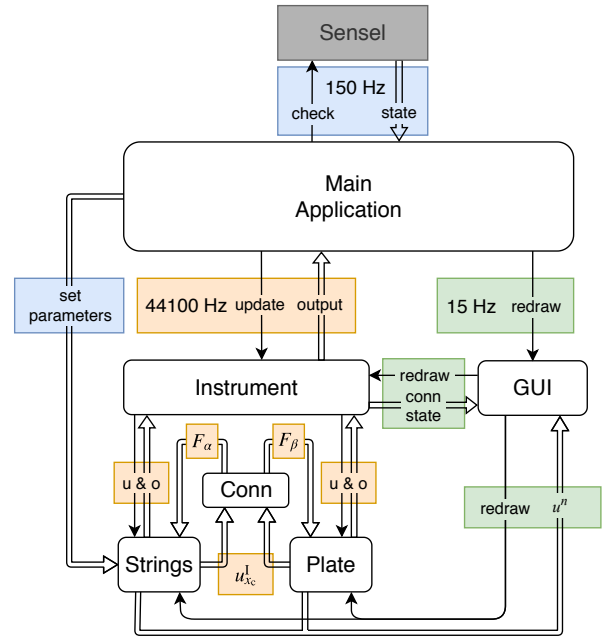


Figure 4. System architecture flowchart. See Section 4.4 for a thorough explanation.

which in turn updates the FDSs in its strings and plate. After the FDS update is done, the intermediate state at the connection points  $u_{x_c}^l$  (where  $x_c = l_{c,\alpha}$  for the string or  $x_c = (l_{c,\beta}, m_{c,\beta})$  for the plate) are sent to the connection (Conn) class which calculates the force-functions  $F_\alpha$  and  $F_\beta$ . These values are then sent back to the string and plate classes and added to their respective states after which their outputs (o) (at arbitrary points) are sent back to the main application. See Algorithm 1 for this ‘order of calculation’.

## 5. INSTRUMENTS AND USER INTERACTION

In this section, the Sensel Morph (or simply Sensel) and user interface will be described in more detail. Furthermore, several configurations of strings, plates and connections that are inspired by real-life instruments will be presented. A demonstration of one of the instruments can be found in [22].

### 5.1 Sensel Morph

The Sensel Morph is a highly accurate touch controller that senses position and force of objects [17] (see Figure 5). We use the Sensel as an expressive interface for interacting with the instrument configurations. Right above the touch-sensitive area, the Sensel contains an array of 24 LEDs that can be controlled from the application.

### 5.2 User interface

Strings are shown as coloured paths (see Figure 6 for a descriptive visualisation). The state  $u^n$  of the string is visualised using the vertical displacement of the paths. Bowed strings are shown in cyan on the top left. The bow is shown

```

while application runs do
  for all elements do
    calculate intermediate state  $\mathbf{u}^I$  using previous
    state values (as in Equation (11))
     $\mathbf{u}_s^I = 2\mathbf{u}^n - \mathbf{u}^{n-1} + K\mathcal{F}$ 
  end
  if element is excited/bowed then
    calculate excitation term  $\mathbf{E}$  and add to interme-
    diate state of the element
     $\mathbf{u}_{s+e}^I = \mathbf{u}_s^I + \mathbf{E}$ 
  end
  for all connections do
    calculate connection forces and add connec-
    tion term  $\mathbf{C}$  to elements to obtain the state at
    the next time step
     $\mathbf{u}_{s+e+c}^{n+1} = \mathbf{u}_{s+e}^I + \mathbf{C}$ 
  end
  update state vectors
   $\mathbf{u}^{n-1} = \mathbf{u}^n$ 
   $\mathbf{u}^n = \mathbf{u}_{s+e+c}^{n+1}$ 
  increment time step
   $n++$ 
end

```

**Algorithm 1:** Pseudocode showing the correct order of calculation. The subscripts for state vector  $\mathbf{u}$  shows what it consists of ('s' for previous state, 'e' for excitation and 'c' for connection).

as a yellow rectangle and moves on interaction. The fretting position is shown as a yellow circle. Plucked strings are shown in purple in the top right, underneath which the sympathetic strings are shown in light green. The plate is shown in the bottom using a grid of rectangles (clamped grid points are not shown). Its state is visualised using a grey-scale. Furthermore, connections are shown using orange circles/squares for the points of connection and dotted lines between these points. Lastly, all parameters that are controlled by the mouse such as output-level and plate-stiffness are located in a column on the right side of the application.

### 5.3 Instruments

We subdivide string-elements into three types: bowed, plucked and sympathetic strings. All strings will be connected to one plate acting as an instrument body of which the user can control the plate-stiffness. Furthermore, the user can change the output-level of each element type. Apart from these parameters, which are controlled by the mouse, the instruments are fully controlled by two Sensels. The instruments we have chosen as our inspiration are the sitar, the hammered dulcimer and the hurdy gurdy.

#### 5.3.1 Bowed Sitar

The sitar is originally an Indian string instrument that has both fretted strings and sympathetic strings. Instead of plucking the fretted strings, we extended the model to bow them. Our implementation consists of 2 bowed strings

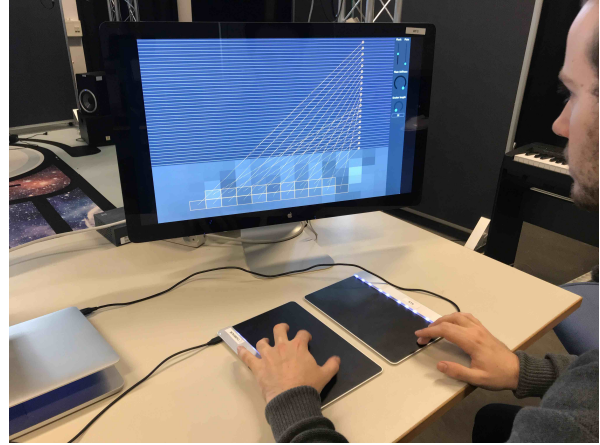


Figure 5. Player using the Sensel Morphs to interact with one of the instruments.

(tuned to A3 and E4), 13 sympathetic strings (tuned according to [23]) and 5 plucked strings (tuned A3-E4 following an A-major scale) as it is also possible to strum the sympathetic strings. See Figure 6 for a visual of the implementation. One Sensel is vertically subdivided into two sections; one for each bowed string. The first finger registered by the Sensel is mapped to a bow and the second is mapped to a fretting finger controlling pitch. The horizontal position of both fingers is visualised using the Sensel's LED array. The frets are not implemented as such (the pitch is continuous), but they are visualised for reference. The horizontal position of the first finger is mapped to the bowing position on the string, the vertical velocity to the bow velocity  $v_B$  and the finger force is linked to the excitation function  $F_B$  (both in Equation (3a)). The other Sensel is subdivided into 5 sections mapped to the plucked strings. These sections are visualised by the LED array for reference.

The mass ratio for the bowed/plucked string to plate connections has been set to  $\mathcal{M} = 2$  and ratio for the sympathetic string to plate connections has been set to  $\mathcal{M} = 0.5$  to increase the effect that the playable strings have on the sympathetic strings.

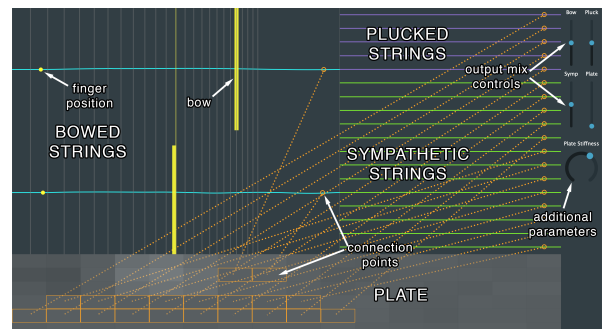


Figure 6. The bowed sitar application. The descriptions of the different elements and other objects are shown in the image, but will (naturally) not be visible in the application.

### 5.3.2 Hammered Dulcimer

The hammered dulcimer is an instrument that can be seen as an ‘open piano’ where the musician has the hammers in their hand. Just like the piano, the strings are grouped in pairs or triplets that are played simultaneously. In our implementation, we have 20 pairs of plucked strings. Even though most hammered dulcimers have more strings, we decided that this configuration has the highest number of strings while maintaining playability. One of each pair is connected to the plate which slightly detunes it, creating a desired ‘chorusing’ effect. See Figure 7 for a visual of the implementation. In order for the excitation to more resemble a strike of a hammer than a pluck, the contents of the cosine in (26) will be multiplied by 2 for the excitation to have a less abrupt ending, something desired for a hammered interaction. Moreover, the excitation-length can be changed to simulate short and long hammer-times.

The Sensels are placed vertically next to each other (see Figure 5). The pair with the lowest frequency will then be located in the bottom right and the highest in the top left, as in the real instrument. As with the plucked strings of the bowed sitar, the LED array is used to visualise the way that the Sensel is subdivided, which is especially useful here as one Sensel controls 10 string-pairs.

The mass ratio is set relatively high ( $\mathcal{M} = 100$ ) to amplify the non-linear interaction between the strings and the detuning of the strings connected to the plate.

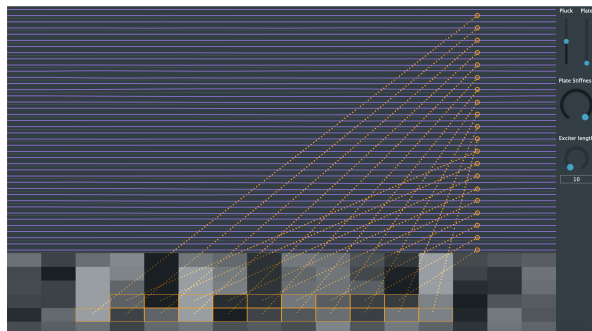


Figure 7. The hammered dulcimer application.

### 5.3.3 Hurdy Gurdy

The hurdy gurdy is an instrument that consists of bowed and sympathetic strings. The bowing happens through a rosined wheel attached to a crank and bows these strings as the crank is turned. It is possible to change the pitch of a few bowed strings - the melody strings - using buttons that press tangent pins on the strings at different positions. The other strings, referred to as drone strings, are mostly tuned lower than the melody strings and provide the bass frequencies of the instrument. The musician can place the bowed strings on rests that keep the wheel from interacting with it.

Our implementation consists of 5 bowed strings subdivided into 2 drone strings tuned to A2, E3 and 3 melody strings tuned to A3, E4 and A4 and 13 sympathetic strings tuned the same way as the sympathetic strings in bowed

sitar. Furthermore, the mass ratios have been set the same as in the bowed sitar application. See Figure 8 for a visual of the implementation.

The Sensel is vertically subdivided into 5 rows that control whether the strings are placed on the wheel. The bowing velocity is mapped to the average pressure of the fingers. The fundamental frequency (in the model  $\gamma/2$ ) of the melody-strings is changed by a Sensel with a piano-overlay acting as a midi controller. A demonstration of this instrument can be found in [22]. It is interesting to note here that the sympathetic strings that are in tune with the harmonics of the bowed strings resonate most, which is expected to happen in the real world as well.

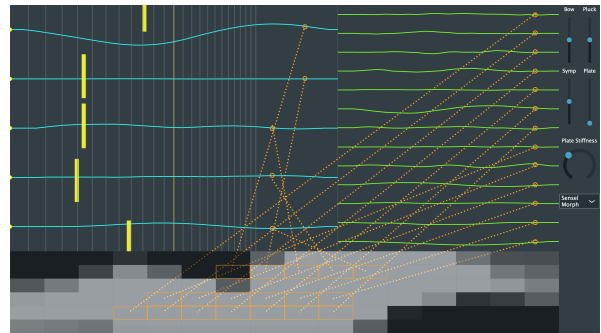


Figure 8. The hurdy gurdy application.

## 6. RESULTS AND DISCUSSION

Table 1 shows the CPU usage (on the same MacBook Pro 2.2 GHz i7 as described before) for the three instruments presented in the previous section. As the Sensel thread contributes a negligible amount to the CPU usage, this is not shown in the table.

Application	No Sound	No Graphics	Total
Bowed Sitar	32	63	85
Dulcimer	30	66	85
Hurdy Gurdy	28	58	78

Table 1. CPU usage (in %) for the instruments found in Section 5. Values show usage of one (virtual) thread and have been taken as an average (with a margin of ~5%) over a short period of time. The two middle columns show usage when the sound or graphics thread has been turned off.

As can be seen from the table, all instruments use about the same amount of CPU and none of them have audible dropouts (CPU < 100%). It can be observed that the graphics use about 20% of the CPU, indicating that there is still much room to increase the complexity of the instrument-configurations before dropouts will occur. On the other hand, should the instruments be used in parallel with other audio applications or plug-ins, the CPU usage has to be greatly reduced. The first step towards this would be to vectorise the FDSs using AVX instructions.

While our instruments have been not formally evaluated yet, we have performed some qualitative evaluations with



sound and music computing experts. The goals of the evaluations were to explore the playability of the instrument, sonic quality and intuitiveness of control. These evaluations showed that especially the bowing interaction feels intuitive and creates a natural sound. The overall sound of the instruments was generally judged to be interesting, but not “sounding like a real-life instrument”. This makes sense, as we did not seek to perfectly model each instrument, but rather used them as an inspiration for the configurations of the physical models. The next step for sound quality would be to replace the thin plate with a more realistic element, such as a wooden instrument body.

## 7. CONCLUSION

In this paper, a real-time modular physical modelling synthesis environment structured as a network of connected strings and plates has been presented. Several instruments have been created in the context of this environment which can be played by a pair of Sensel Morphs allowing for highly expressive control of these instruments. Informal evaluations with professional musicians have confirmed that the interaction is found natural and the output sound interesting. Further steps to improve this project are to optimise the algorithm and to replace the plate with a more realistic instrument body.

## Acknowledgments

This work is supported by NordForsk’s Nordic University Hub Nordic Sound and Music Computing Network Nordic-SMC, project number 86892.

## 8. REFERENCES

- [1] C. Cadoz, “Synthèse sonore par simulation de mécanismes vibratoires,” 1979, thèse de Docteur Ingénieur, I.N.P.G. Grenoble, France.
- [2] C. Cadoz, A. Luciani, and J.-L. Florens, “Responsive input devices and sound synthesis by simulation of instrumental mechanisms,” *Computer Music Journal*, vol. 8, no. 3, pp. 60–73, 1983.
- [3] C. Cadoz, A. Luciani, and J. L. Florens, “Cordis-anima: a modeling and simulation system for sound and image synthesis: the general formalism,” *Computer music journal*, vol. 17, no. 1, pp. 19–29, 1993.
- [4] J. D. Morrison and J.-M. Adrien, “Mosaic: A framework for modal synthesis,” *Computer Music Journal*, vol. 17, no. 1, pp. 45–56, 1993.
- [5] J. O. Smith, “Physical modeling using digital waveguides,” *Computer music journal*, vol. 16, no. 4, pp. 74–91, 1992.
- [6] P. Ruiz, “A technique for simulating the vibrations of strings with a digital computer,” Master’s thesis, University of Illinois, 1969.
- [7] L. Hiller and P. Ruiz, “Synthesizing musical sounds by solving the wave equation for vibrating objects: Part I,” *Journal of the Audio Engineering Society*, vol. 19, no. 6, pp. 462–470, 1971.
- [8] —, “Synthesizing musical sounds by solving the wave equation for vibrating objects: Part II,” *Journal of the Audio Engineering Society*, vol. 19, no. 7, pp. 542–550, 1971.
- [9] R. Bacon and J. Bowsher, “A discrete model of a struck string,” *Acustica*, vol. 41, pp. 21–27, 1978.
- [10] A. Chaigne, “On the use of finite differences for musical synthesis. Application to plucked stringed instruments,” *Journal d’Acoustique*, vol. 5, no. 2, pp. 181–211, 1992.
- [11] A. Chaigne and A. Askenfelt, “Numerical simulations of struck strings. I. A physical model for a struck string using finite difference methods,” *Journal of Acoustical Society of America*, vol. 95, no. 2, pp. 1112–1118, 1994.
- [12] S. Bilbao, *Numerical Sound Synthesis, Finite Difference Schemes and Simulation in Musical Acoustics*. John Wiley and Sons, Ltd, 2009.
- [13] —, “A modular percussion synthesis environment,” *Proc. of the 12th Int. Conf. on Digital Audio Effects (DAFx-09)*, 2009.
- [14] S. Bilbao, A. Torin, P. Graham, J. Perry, and G. Delap, “Modular physical modeling synthesis on gpu,” in *Proceedings of the 40th International Computer Music Conference and the 11th Sound and Music Computing Conference*, 2013.
- [15] C. Webb and S. Bilbao, “On the limits of real-time physical modelling synthesis with a modular environment,” *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, 2015.
- [16] K. Franinović and S. Serafin, *Sonic interaction design*. MIT Press, 2013.
- [17] Sensel Inc. (2018) Sensel morph. [Online]. Available: <https://sensel.com/>
- [18] C. Desvages and S. Bilbao, “Two-polarisation physical model of bowed strings with nonlinear contact and friction forces, and application to gesture-based sound synthesis,” *Applied Sciences*, 2016.
- [19] K. Graff, *Wave Motion in Elastic Solids*. New York, New York: Dover, 1975.
- [20] JUCE ROLI. (2019) JUCE. [Online]. Available: <https://juce.com/>
- [21] J. Wallis, *A treatise of algebra, both historical and practical*. London, 1685.
- [22] S. Willemsen. (2019) Hurdy gurdy demo. [Online]. Available: <https://www.youtube.com/watch?v=BkxLji2ap1w>
- [23] Joe Rizzo. (2015) How to tune a sitar. [Online]. Available: <http://www.joerizzo.com/sitar/>

# An Interactive Music Synthesizer for Gait Training in Neurorehabilitation

**Prithvi Kantan**

Sound and Music Computing  
Aalborg University, Copenhagen  
pkanta18@student.aau.dk

**Sofia Dahl**

Dept. of Architecture, Design and Media Technology  
Aalborg University, Copenhagen  
sof@create.aau.dk

## ABSTRACT

Rhythm-based auditory cues have been shown to significantly improve walking performance in patients with numerous neurological conditions. This paper presents the design, implementation and evaluation of a gait training device capable of real-time synthesis and automated manipulation of rhythmic musical stimuli, as well as auditory feedback based on measured walking parameters. The proof-of-concept was evaluated with six healthy participants, as well as through critical review by one neurorehabilitation specialist. Stylistically, the synthesized music was found by participants to be conducive to movement, but not uniformly enjoyable. The gait capture/feedback mechanisms functioned as intended, although discrepancies between measured and reference gait parameter values may necessitate a more robust measurement system. The specialist acknowledged the potential of the gait measurement and auditory feedback as novel rehabilitation aids, but stressed the need for additional gait measurements, superior feedback responsiveness and greater functional versatility in order to cater to individual patient needs. Further research must address these findings, and tests must be conducted on real patients to ascertain the utility of such a device in the field of neurorehabilitation.

## 1. INTRODUCTION

This paper presents a novel application capable of measuring gait parameters and delivering interesting, time-evolving auditory stimuli based on gait quality for rehabilitation purposes. The primary goal is to increase engagement and enjoyment of therapy, improving patient motivation and adherence to frequent therapy, thereby leading to more favorable clinical outcomes. Brain damage from disease, infarction or infection frequently compromises gross motor function, resulting in impairments to essential activities like walking. Gait (walking) quality and mobility are important predictors of survival [1], cognitive decline [2], fall risk and perceived quality of life among older adults [1]. Besides age-related deficits, neurological conditions such as *Parkinson's Disease (PD)*, stroke, *Acquired Brain Injury (ABI)* and others have the capability to destroy gait function in an either acute or chronic manner. Prompt

and regular rehabilitation has been found to be a critical determinant of long-term deficits [3]. While exercise helps preserve physical function, exercise protocols are typically not readily accessible in homes [4] and novel cost-effective rehabilitation strategies are needed [5]. In this context, the auditory modality can be advantageous over the visual and haptic ones in terms of hardware requirements and computational burdens [6]. Moreover, music-based interventions are being increasingly studied [7] and are attractive in that they can heighten enjoyment during exercise and, in turn increase exercise adherence [2].

Given the ability of rhythmic music to motivate humans and induce bodily movement [8], we propose a gait training system generating evolving musical stimuli in real-time, as well as spontaneous auditory feedback based on measured walking performance. The unique contribution lies in the direct influence of walking quality on the behavior of discrete entities within the composite auditory stimulus. Equally critical is an interface that is simple and intuitive enough for operation by a therapist, and versatile enough to tailor stimuli to a diversely afflicted patient group. The gait measurements collected are stored after each session to provide valuable information on patient progress. In the following sections we will discuss related research and present the design and implementation of multiple cohesively interacting systems for gait data acquisition, analysis and audio synthesis. As evaluation, the device was tested with six normal-walking individuals and critically reviewed by one neurorehabilitation specialist.

## 2. RELATED WORK

### 2.1 Rhythmic Auditory Stimulation (RAS)

*Rhythmic Auditory Stimulation (RAS)* is a rehabilitation technique of rhythmic motor cuing to facilitate training of movements that are intrinsically and biologically rhythmic, such as walking. RAS has been used in the rehabilitation of patients suffering from strokes [9], PD [10], ABI [11], and several other neurological conditions [12]. Essentially, it is the application of a rhythmic pulse (or beat) to organize periodic bodily movements in a process that occurs below conscious perception and functions to improve movement efficiency. The pulse often takes the form of a metronome click, or rhythm-based music. In PD [4] and stroke rehabilitation [9], RAS has been shown to improve numerous gait performance parameters [13]. RAS efficacy may depend on individual characteristics, disease severity and impaired beat-synchronization ability [12]. The

Copyright: © 2019 Prithvi Kantan et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

beneficial effects of RAS reverse themselves over time if therapy is not adhered to [13]. Different individuals have different preferred movement tempi in general [2, 7] and the degree of entrainment drops significantly if the cuing tempo is over 2.5% greater or 3% less than the preferred tempo [14]. The daily administration duration of RAS is 10-30 mins, once or twice. The frequency of administration depends largely on patient endurance. A tempo increase of 5% is attainable without compromising normal walking patterns.

## 2.2 Movement Sonification

Sonification may be defined as the transfer of data and data relationships into non-speech audio for communication and interpretation [15]. In the rehabilitation context, the main advantage of sonification is its ability to enhance self-awareness of physiological processes and physical motion. Sonification makes it possible to be cognizant of, and control motor performance output parameters that efface themselves from conscious experience in most behavior [14]. The goal of a sonification-based system is to make distinct performance parameters explicit in corresponding auditory biofeedback systems. The 3Mo model proposed by Maes et. al. [14] suggests the use of musical biofeedback due to the potential that music has, to *motivate* physical activity, *monitor* physiological processes and *modify* these processes. A good example of this is the robotics-based system developed by Zanotto et. al [16] in which hip and knee angles during gait were sonified in real-time using formant synthesis. Previous studies have demonstrated that music listening can activate the human reward system. In line with the idea of reward-based reinforcement learning, Maes et. al. argue that pleasant and rewarding states promoted by music may function as an attractive force of motor behavior. Reward and punishment are hence considered constraints, guiding motor behavior to specific goals [14]. Optimal sonification strategies associate *wanted* motor behaviors to *pleasant* auditory states and vice versa. Musical expressiveness, novelty and surprise, along with tension and uncertainty are important elements for the sustainment of reward responses [14]. Auditory feedback may be successfully used in the rehabilitation context because it can be perceived without requiring patients to pay attention to a screen, and can be processed with relatively little cognitive effort [6].

## 2.3 Measures of Gait Performance

Human gait involves alternating sequences in which the body is supported first by one limb, which is contacting the ground, and then by the other [17]. For each limb, the period of support is referred to as the *stance phase*, and that of non-support is the *swing phase*. These events are separated by the instants at which the foot contacts and leaves the ground, and gait cycles are usually defined relative to these instants. A more comprehensive overview of the subject is presented in [18]. One approach to gait measurement involves a broad structural group of parameters that captures both spatiotemporal and dynamic characteristics. Lord et. al. [19] describe a 5-domain concep-

tual model. They identified 16 core variables explaining 84.6% of the variance between controls and 121 PD patients, which inform the measurement mechanisms of our application. Currently, only temporal parameters are considered, namely *step time*, *stance time* and *swing time*, as well as their *temporal variability* and *asymmetry*.

## 2.4 Applications for RAS-based gait rehabilitation

In recent years, some technological applications targeting gait rehabilitation based on RAS principles have been developed. The IM Gait Mate is a therapy modality to assess and treat motor planning, sequencing, coordination and balance [20]. The device targets patients suffering from PD, spinal cord injury, ABI and other related conditions. Wireless insoles are inserted in the patient's shoes to detect heel-strikes. The patient hears a beat through wireless headphones or speakers and is asked to match their *cadence* (*steps per minute*) to the tempo provided. Real-time speech-based audio feedback is provided related to step rate, dictated by how closely the cadence matches the auditory stimulus. A slightly different approach is used in D-Jogger, an interactive music player that aligns recorded music to the user's gait [21, 22]. Rather than asking the user to match their cadence to the music, D-Jogger adjusts the tempo of the music so that each beat coincides with a footfall. User cadence is determined in real-time using sensors, and the system automatically selects a song with similar tempo and continuously adjusts it to match cadence in an imperceptible fashion. If the user cadence changes markedly, the system switches to a different song.

The D-Jogger and similar systems are advantageous in situations where spontaneous gait synchronization does not occur [22], and can be categorized as *closed-loop* where the stimulus tempo adjusts itself to the user's cadence [12]. Conversely the IM Gait Mate would be an *open-loop* system. For both IM Gait Mate and D-Jogger the level of interaction between the user and the stimulus itself is quite limited, given the pre-recorded nature of the stimuli. Furthermore, only cadence is measured, limiting their ability to capture finer-grained gait impairments. We argue that the dynamic generation of evolving rhythmic music based on several dimensions of gait quality would be more motivating for rehabilitation, and versatile enough for useful administration to multiple patient groups.

## 3. DESIGN AND IMPLEMENTATION

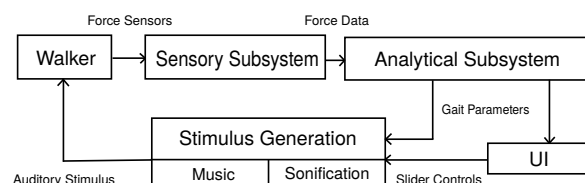


Figure 1. Block diagram displaying the overall system with its subsystems.

The design of the application is informed by prior re-

search, and must fulfill the following requirements:

- Real-time generation of activating and motivating musical stimuli.
- Sonification of important gait parameters to provide real-time auditory feedback having customizable intensity.
- Non-invasive measurement and storage of clinically relevant gait parameters, for the monitoring of patient performance and progress.

The application is designed as a combination of functionally distinct but highly interdependent and cohesive subsystems, illustrated in Figure 1 and described as follows:

### 3.1 Sensory Subsection

The sensory subsystem is the data-acquisition system targeted towards gait measurement. The approach used is that of foot-based FSR sensors (*force-sensitive resistor*). The primary hardware used is the Trigno EMG System by Delsys™, which consists of a USB-controlled Base Station unit and a wireless Trigno 4 Channel FSR Sensor™ with an operating range of 20m. The Base Station acts as a receiver, and force data is transmitted to the application via TCP/IP [23]. Two force data channels are captured, from a 15 mm<sup>2</sup> FSR membrane on each heel. The signals are sampled at 148 Hz, and quantized to 10 bits per sample.

### 3.2 Gait Feature Extraction

The analytical subsystem handles the real-time extraction of clinically important gait features from the force time series supplied by the sensory subsystem. It performs foot-step detection, stance/swing detection as well as gait parameter calculation and storage. It relays these values to both the user interface and the control channels of the stimulus-generation subsystem, effectively acting as the central information hub of the application. It is implemented in C++ using a JUCE timer to fetch new force samples for periodic analysis.

#### 3.2.1 Footstep Detection

Given the implicit periodicity of walking, heel force contours appear as a series of evenly spaced amplitude fluctuations, corresponding to the support duration of that foot. Each step period can be measured as the interval between two contour points in the same phase, or simply two contour maxima. However, since the force variations across steps are neither smooth nor identical; step peaks often appear spiky with multiple local maxima per support period. Therefore, second-order IIR Butterworth lowpass filters with their -3dB frequency set at 0.5 Hz are used to smooth the force contours prior to peak detection. Peaks exceeding 14% of the maximum force range are detected as valid steps. The phase delay incurred by the IIR filter causes heel-strikes to be detected approximately 300 ms after they occur, delaying all gait parameter calculations as a result. A more recent perspective views the smoothing filter as a capacitor which *accumulates* force while foot contact is maintained. Correspondingly, the local minimum preceding each of the maxima may be detected as the

instant of foot-contact, and the maxima themselves represent instants of non-contact. This approach neutralizes the filter phase delay.

#### 3.2.2 Stance and Swing Detection

The unfiltered force time series of each foot is segmented into stance and swing phase by simple thresholding, at an empirically determined level of 20% of full-range force, to prevent false detections due to noise.

#### 3.2.3 Parameter Calculation and Storage

The duration of each step, stance and swing period is calculated in real-time post detection and stored in separate vectors. As motivated in Section 2, the *mean-normalized variability* and *L-R asymmetry* for each of these (along with average stance/swing ratio) are recalculated with every newly completed step, at two different timescales:

**Long Term:** This timescale spans the entire training session from the first detected step. The trajectory of long-term measurements across multiple training sessions can be used by therapists to assess *improvement or deterioration* in patient gait.

**Short Term:** The same gait parameters are computed over an empirically determined window of only the five most recent steps, thus more numerically sensitive to new measurements. These are input to the control channels of the stimulus-generation subsystem for *sonification* purposes.

### 3.3 Stimulus Generation Subsystem

This subsystem generates and manipulates auditory stimuli for gait entrainment. This process involves the sequencing, arrangement and expressive interpretation of time-evolving musical layers that culminate in a well coordinated ensemble of rhythmic instrumental music. Also, the synthesizer sonifies gait performance, for which it monitors specific short term parameters and modifies the stimulus accordingly. Stylistically, the music is closest to the electronic dance music genre, which has been found to be most conducive to movement in related studies [8]. The synthesizer itself is implemented in FAUST (Functional Audio Stream), which is an audio domain-specific functional programming language. Although there exists a wealth of easily available high-quality music loops, the real-time synthesis approach is attractive due to its potential for fine parameter control and overall sonic versatility. The Faust2Api library was used to create a JUCE-compatible C++ class for the synthesizer, enabling communication with the analytical subsystem. It also allows direct user manipulation of synthesizer parameters from the same interface that displays gait parameter values, allowing for convenient operation and monitoring.

#### 3.3.1 Structure of Musical Content

The core ensemble consists of typical percussive elements found in electronic music, as well as multiple pre-composed melodies that reinforce the underlying rhythm. The time signature is 4/4 throughout, and the tempo is user-adjustable, depending on the preferred cadence of the



Track no.	Instrument	Basic Excitation	Synthesis Method	Bandwidth (Hz)	Effect Chain
1	Bass Drum	Sine Sweep	Subtractive	60-200	3dB Boost @ 70 Hz
		White Noise	Subtractive	1500-5200	Cubic Soft Clipper
2	Snare Drum	White Noise	Subtractive	100 - 8000	-
3	Hi-Hat	White Noise	Subtractive	10000 - 16000	-
4	Crash Cymbal	White Noise	Subtractive	9000 - 20000	-
5	Bass Synth	Sawtooth	Subtractive	50 - 200	8 dB Boost @ 110 Hz
6	Bass Staccato	Sine	FM	150 - 2000	-
7	Main Melody	Sine	FM	f0 - 1000	Dotted Echo, Haas Delay, Hard Clip
8	Secondary Melody	Sine	FM	f0 - 5000	-

Table 1. Synthesis methods and effect chains of each instrument in the ensemble ('f0' refers to the note fundamental frequency and FM stands for Frequency Modulation).

walker. The underlying rhythmic pattern remains uniform throughout, and musical variation is realized in the manipulation of secondary rhythms and melodic patterns. The musical characteristics are designed to match those found in [8] to be the most activating in terms of walking vigor.

### 3.3.2 Clocking and Musical Timekeeping

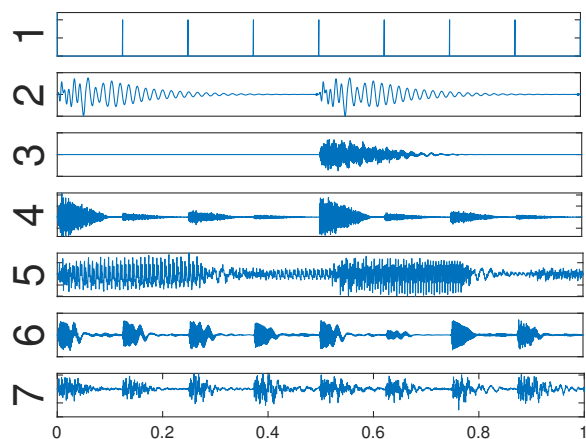


Figure 2. An illustration of how the Master Clock (1, top panel) serves as a triggering mechanism for some of the instruments (2-7) described in Table 1.

At the core lies a continuous, isochronous impulse train, whose frequency is governed by the externally configured tempo. This acts as the primary clocking and triggering mechanism and orchestrates the synchronized playback of every instrument. Its frequency is precisely four times the beat interval to enable the use of eighth and sixteenth note subdivisions in the music. External changes to the tempo alter the impulse train frequency, in turn altering the triggering rate of all instruments at once. This *master clock* is generated using a FAUST impulse train function. For musical time-keeping, there are counters monitoring the master clock to count elapsed measures and the current position within a measure. Their state is referenced during au-

dio synthesis to determine whether to mute a track, audio effect automation, and instantaneous melody note parameters.

### 3.3.3 Introduction of New Instruments

Effort of walking is rewarded regardless of gait performance. The total *footstep count* indicates the physical work performed by a patient. The music commences with only the bass drum and the bass synth and each new instrument is added as its minimum step count condition is satisfied (snare drum, hi-hat, etc.). Boolean variables in FAUST take values of either 0 or 1, enabling step count conditions to double as instrument on/off switches. To gently fade each new instrument in as its step condition is satisfied (rather than a discontinuous and un-musical 0-to-1 transition), the Boolean variables are smoothed using a one-pole filter with a 4 second integration time. This filtering helps achieve a gradual gain increase from the instant the step condition is satisfied.

### 3.3.4 Percussion Synthesis

Most percussive rhythm patterns may be viewed as periodic arrangements of distinct impulsive sounds in a specific temporal order. The mechanism for percussion synthesis is a filtered steady-state excitation multiplied by a temporal envelope triggered by the master clock. The triggering occurs at sub-multiples of the clock frequency depending on the rhythm. An example of this is the bass drum, whose envelope is triggered every four clock pulses, or more precisely, triggered by 'the first of every four pulses'. It therefore plays on every beat. In contrast, the snare drum envelope is triggered by the fifth of every eight pulses, thus playing on the second and fourth beats of a measure (visualized in Figure 2). A large variety of rhythms can thus be easily generated. The excitation and envelope parameters vary among the percussion instruments to achieve the desired timbres. All parameters were iteratively determined using an analysis-by-synthesis method.

### 3.3.5 Melody Synthesis

The paradigm employed is inspired by MIDI-based symbolic notation, in which note/velocity information is stored in relative music time. This information may then be reproduced by the synthesizer. In the current context, note number, on/off, and velocity for each melody are stored chronologically in look-up tables. Counters deriving from the master clock keep track of the number of elapsed 8th and 16th notes in the current measure. The instantaneous state of these counters is used to iterate through the look-up tables, fetching note information for each instrument. The output is a specific timbre with the required instantaneous fundamental frequency. The note on/off table specifies note onset positions within a bar, effectively serving as a trigger condition for envelope generation. Because note triggering is orchestrated by the master pulse counters, melody tempo and note length vary proportionally with the master tempo, ensuring grid-locked synchronization among all instruments. This is important in a scenario where timing precision is paramount. Overall mix clarity, transient impact and spectro-temporal separation between individual timbres are critical to the user experience. Indistinct, unclear or harsh sound can quickly become fatiguing and unpleasant, which is undesirable for training sessions over ten minutes in length. Therefore, special attention is paid to the interaction of sound sources, leveraging modern production strategies to deliver a well-balanced stereo mix. Table 1 offers a more detailed description of the synthesis methods of all musical instruments.

### 3.3.6 Sonification Synthesis

Aside from the generation of the musical elements, the Stimulus Generation subsystem also handles the sonification of various measured gait parameters. The sonification philosophy is as advocated by Maes et. al. [14], with the *addition of unpleasant stimuli, or detrimental modification of the existing music stimuli*, to 'punish' sub-optimal gait performance. In practice, this is achieved by mapping a subset of the short-term gait parameters to 'user sliders' on the synthesizer program, that control the intensity of each sonification type. Thus, the intensity is a continuous quantity, and varies in real-time with the temporal gait parameter mapped to it. The interface has a slider to scale the overall strength of punishment to optimize usability for differently severe impairments.

The applied sonifications are described as follows:

**Rhythm Salience:** The reduction of *step-time variability* through rhythmic entrainment is an important outcome of RAS therapy. Beat clarity is critical to effective entrainment, and the primary beat is largely carried by the percussion instruments. An increase in short-term step time variability causes an increase in the relative level of the percussive instruments with respect to the melody instruments. The *punishment* lies in the resulting attenuation of melodic content in the mix. This also serves to strengthen entrainment, ultimately improving step regularity.

**Annoyance Notes:** The analytical subsystem compares the measured average stance/swing time ratio with

the documented ratio for normal walking - 1.61 [24]. A large discrepancy indicates sub-optimal support time distribution and is sonified in the form of random-frequency annoyance notes from a sawtooth wave generator. The intensity is directly proportional to the squared deviation from the normal stance/swing ratio. The triggering frequency of these notes is dictated by the master clock, so that they do not affect rhythm perception.

**Melody Detuning:** Increases in *Swing Time Asymmetry (STA)* are sonified by directly mapping it to the depth control of a ring modulator in the signal path of the main melody synth. The modulation frequency is not in the key of the music, so the effect of increased STA is increased dissonance in the reproduced melody. Symmetric walking translates to a very low asymmetry quotient, and therefore negligible modulation.

**Noise:** Short-term increases in *Swing Time Variability* are sonified using a white noise generator, punishing increasing variability by increasing noise intensity. The noise is processed with a high-intensity flanger with tempo-dependent sweep rate, giving it a rhythmic quality.

## 4. EVALUATION

To ensure that the application exhibited both the expected sonic behavior and gait measurement ranges, a walking test was conducted on unimpaired, healthy participants. Aside from this experimental evaluation, the application was demonstrated to a neurorehabilitation specialist to assess its utility as a rehabilitation tool.

### 4.1 Experiment

#### 4.1.1 Participants

Six individuals (one female, mean age 24.8 years) with no documented neurological conditions or gait disorders volunteered themselves as participants for this study. The participants all had a prior music background and were students at Aalborg University. Informed consent was obtained, and refreshments were provided as compensation.

#### 4.1.2 Experimental setup

The experiment took place in a large, quiet room with the measurement system set up in the center of a roughly circular demarcated walking track. The music was played back at a comfortable level over a set of full-range stereo loudspeakers. Due to the cyclic nature of the track, the participants heard the music at a roughly consistent loudness level throughout their traversal.

#### 4.1.3 Procedure

Each participant individually tested the system in two phases. The first phase was a trial run to determine their preferred cadence. Once securely fitted with the sensor apparatus, they were instructed to walk freely, at a comfortable pace along the path. The music was played, and the tempo was adjusted manually until it matched the gait of

Gait Parameter	Measured	Reference
Mean Step Time (ms)	539 (2)	537(47)
Step Time Variability (%)	9.68 (1.07)	3.05 (1.10)
Mean Stance Time (ms)	599.7 (60.70)	688 (72)
Mean Swing Time (ms)	366 (61)	386 (30)
Stance/Swing Time Ratio	1.695 (0.414)	1.782 (0.188)
Stance Time Asymmetry (%)	12.30 (7.76)	1.29 (1.35)
Swing Time Asymmetry (%)	7.20 (2.97)	2.30 (2.43)
Stance Time Variability (%)	12.30 (4.50)	2.87 (1.16)
Swing Time Variability (%)	43.15 (41.37)	3.93 (1.42)

Table 2. Measured gait parameter values with reference values derived from [25], shown as Mean and Standard Deviation SD (in parenthesis).

the participant. After this calibration phase, the new tempo was initialized, sensor measurement was commenced, and the participant was signaled to begin walking from a designated starting point. The instruction this time was for footsteps to be actively timed to the music. Feedback sonification was enabled at the nominal intensity level without participants being informed of what it was. The duration of each trial spanned the time taken by the participant to complete 400 steps, and this was dependent on individual preferred cadence. The listed long-term temporal gait parameters were automatically measured and systematically stored at the end of the 400th step, simultaneously concluding the experiment. Following each trial, participants were interviewed in structured fashion, and key questions were put forth regarding the distinct aspects of the experiment. They were asked to rate the comfort and freedom of movement (from 1, not at all comfortable/very low freedom to 5, very comfortable/very high freedom) while wearing the apparatus. Pertaining to the music, key questions concerned appropriateness of tempo, beat clarity, enjoyability, musical evolution, and conduciveness to movement. They were asked if they noticed any unusual sounds, and if so, what their impression was of them.

#### 4.1.4 Results

All participants reported the task simple and the pre-calibrated tempo easy to walk to. The musical beat was clearly perceived by all participants, and temporal evolution in the music was noticed. Four out of six participants found the music encouraging to move to, but half of them did not find the music enjoyable. Four of six noticed sounds that were not part of the music and one found these sounds unpleasant. Mean rating for comfort while donning the sensor apparatus was 3.92 (ranging from 3 to 4.5) while mean rating for freedom of movement was 4.42 (ranging from 4 to 5).

Table 2 shows the measured gait parameters, averaged

across participants and compared to reference values derived from literature [25]. No false step detections were observed in any trial. Measured Mean Step Time showed a high level of agreement with reference figures, and Mean Stance, Swing Time and Stance/Swing Ratio were within range. On the other hand, Long term Asymmetry and Variability measures were significantly exaggerated as compared to reference values, although this discrepancy was *not* observed on the short-term timescale.

## 4.2 Expert Interview

In addition to the gait experiment, the application was demonstrated to a neurorehabilitation specialist by means of a walking test, simulating both normal and impaired gait modalities. The following key questions were put forth and a thorough assessment was obtained.

### *Clinical Role (Target Group and Use Case):*

The specialist stated that the main target group would constitute PD, stroke and ABI patients, and that the most convenient therapy setting would be a treadmill protocol.

### *Main Benefits from a Therapist's Perspective:*

The specialist envisioned the gait measurement and real-time feedback to have potential as novel aids to performance evaluation and patient self-awareness.

### *Detection of Impaired Gait Modalities:*

The inability of the application to evaluate foot roll-over quality (owing to only a single heel sensor) was pointed out by the specialist. He also enumerated several phenomena that cannot be captured by temporal measures alone, such as low gait speed, crouch gait and limb circumduction. The use of accelerometers and force membranes with greater surface area was suggested.

### *Presentation of Auditory Stimuli:*

The specialist pointed out problems with the sonification philosophy of punishing any deviations from parametrically normal gait, mainly rooted in the wide range of pathologies and principal gait problems exhibited by patients. He not only stated the importance of safety and balance in the short term, but also the need for individualized performance baselining and customizable sonification mapping to cater to diverse individuals. He added that the subjective definition of the term *unpleasant* would create ambiguity between the perceptual notions of punishment and reward, especially for cognitively damaged patients. The mapping of gait parameters to auditory manipulations was also not seen as intuitive by the specialist. Lastly, he noted the time-lag between a gait event (eg. an asymmetric stride) and its respective sonification, which would lead to uncertainty and confusion for the patient while experimenting with gait technique, damaging the delicate re-learning process.

## 5. DISCUSSION

The objective of this study was to design an interactive music-based gait training application based on RAS in order to improve patient motivation and adherence to gait therapy. We presented and tested the implementation of the proof-of-concept application, generating a time evolving

musical ensemble controlled by measured temporal gait parameters. It is acknowledged that an important limitation of this study is the lack of clinical trials conducted. Compared to existing systems such as D-Jogger [22] and IM Gait Mate [20], the merits of this application are seen in the detailed calculation of numerous relevant gait parameters apart from cadence and step-time variability, as well as the multidimensional influence of walking quality on the auditory stimulus. These are reinforced by the specialist's acknowledgement of the potential of the application to help individuals from our original target group, as well as the benefits of the gait measurement and real-time auditory feedback mechanisms. In practical terms, the experimental evaluation broadly showed that the application functioned as intended with multiple individuals. The suitability of the stimulus for interactive gait entrainment was corroborated by the perceived intuitiveness of walking to the generated music, the easy discernment of temporal evolution and the conduciveness to movement. High ratings of comfort and freedom encourage the future use of a force-sensor based gait measurement system.

One concern stems from the lack of agreement among participants regarding whether the music itself was enjoyable. Although this disagreement was expected due to the diversity of individual music preferences, it necessitates the design of a music synthesis system with the capability of morphing seamlessly between distinct styles while maintaining its movement-inducing quality. The average age of the target group is also higher than that of the test group used, so it is important that the trends in musical preference of target individuals are studied in greater detail. Because the test group mainly comprised normal walking individuals, there were very few instances where gait sonification was audible in the stimulus, pointing to mostly correct triggering of sonification mechanisms (or lack thereof). However, among the participants who did perceive some sonification effects, the general disposition towards these effects was neutral. Although these were designed to sound unpleasant, this disposition could be ascribed to sonic expectations in the electronic genre, where timbres are inherently noisy and bright, with more tolerated inharmonicity. Alternative sonification strategies must therefore be considered. Firstly, to cater to the wide range of principal gait problems and severity, individualized performance baselining is a necessary addition. A possible sonification alternative is to conceive of sonic reward and punishment purely in terms of musical complexity, such that good gait performance with respect to the baseline is rewarded with more interesting rhythms and melodies, and the opposite effect for deteriorating performance. Additionally, melody and percussion envelopes can be triggered by step onsets, encouraging tight synchronization and giving the user a greater sense of agency and control. This could potentially solve the problems of temporal spontaneity and reward/punishment ambiguity predicted by the specialist. Furthermore, pleasantness and relative discernibility of each sonification strategy must be investigated in more detail through experiments. Discretizing sonification intensity levels based on the measured *just-*

*noticeable difference* could potentially make variations in auditory feedback more explicit.

The next topic is the automated gait parameter measurements. The noted discrepancies between measured values and reference values may have been caused by differences in exact sensor placement, spikes in asymmetry during turning or outliers created by initial shuffled steps. Regardless, the high step detection latency (300 ms) must be addressed; the capacitor analogy described in Section 3.2 has shown promise in initial tests. Temporal spontaneity of gait sonification is critical to the effectiveness of the application, and the short-term measurement window may be shortened to improve this. Additional membranes must be introduced to represent the forefoot for evaluating roll-over. An adjustable measurement prototype must be fabricated to ensure accurate sensor placement and reduce setup time. The initial step measurements must be discarded to obtain more accurate long-term figures for evaluation. Periodic automated cadence detection is also a necessary provision for setting music tempo. The path of traversal should be straight, to prevent turning-related inaccuracies. Accelerometers to capture gait speed and crouch gait, along with the design of a treadmill protocol for testing and therapy have been added to the scope of future studies.

## 6. CONCLUSIONS

The goal of this study was to develop a music-based interactive gait training device for patients suffering from neurological conditions, creating an organic and enjoyable setting capable of improving motivation and adherence to therapeutic exercise. A proof-of-concept was designed and implemented, and subsequent tests and evaluation processes on normal test participants revealed both merits and deficiencies in the auditory presentations. Practical difficulties and inaccuracies in some of the gait measurement mechanisms also came to light. The expert interview provided us with much needed feedback and insight into the rehabilitation process. Taking into account the infancy of the project in its current state, the work carried out here serves as a useful foundation for future investigation driven by the experimental findings. Given the increasing need for affordable and accessible exercise protocols for neurological patients, an interactive device wielding the universal appeal and therapeutic prowess of music may be instrumental in the recovery and maintenance of physical function and mobility among community-dwelling individuals afflicted by debilitating neurological conditions.

## Acknowledgments

We are indebted to the participants in the study; J. Greve and P. Williams for assisting the fabrication of the test apparatus; D. Curtis for his valuable comments in evaluating the application, R. Jakobsen, C. Ringsted, and C. Khadye for suggestions, comments, and feedback; as well as the anonymous reviewers for their helpful comments.

Author Kantan was mainly responsible for designing, implementing, evaluating and writing this paper as part of a student project at the MSc. Program in Sound and



Music Computing, with Dahl as supervisor. The work was partially funded by NordForsk's Nordic University Hub Nordic Sound and Music Computing Network Nordic-SMC, project number 86892.

## 7. REFERENCES

- [1] M. Hirvensalo, T. Rantanen, and E. Heikkinen, "Mobility difficulties and physical activity as predictors of mortality and loss of independence in the community-living older population," *Journal of the American Geriatrics Society*, vol. 48, no. 5, pp. 493–498, 2000.
- [2] B. Pumper, H. Wirkkala, N. Smyth, R. Forkan, M. A. Ciol, and A. Shumway-Cook, "Exercise Adherence Following Physical Therapy Intervention in Older Adults With Impaired Balance," *Physical Therapy*, vol. 86, no. 3, pp. 401–410, 03 2006.
- [3] M. Weinrich, D. C. Good, M. Reding, E. J. Roth, D. X. Cifu, K. H. Silver, R. L. Craik, J. Magaziner, M. Terrin, M. Schwartz, and L. Gerber, "Timing, intensity, and duration of rehabilitation for hip fracture and stroke: Report of a workshop at the national center for medical rehabilitation research," *Neurorehabilitation and Neural Repair*, vol. 18, no. 1, pp. 12–28, 2004.
- [4] A. Ann Clair, K. E. Lyons, and J. Hamburg, "A feasibility study of the effects of music and movement on physical function, quality of life, depression, and anxiety in patients with parkinson's disease," *Music and Medicine*, vol. 4, pp. 49–55, 01 2012.
- [5] A. J. Sihvonen, T. Srkm, V. Leo, M. Tervaniemi, E. Altmüller, and S. Soinila, "Music-based interventions in neurological rehabilitation," *The Lancet Neurology*, vol. 16, no. 8, pp. 648 – 660, 2017.
- [6] K. Franinovic and S. Serafin, *Sonic Interaction Design*. Cumberland MIT Press, 2016.
- [7] M. Thaut, *Rhythm, Music, and the Brain: Scientific Foundations and Clinical Applications*. Routledge, 2013.
- [8] M. Leman, D. Moelants, M. Varewyck, F. Styns, L. van Noorden, and J.-P. Martens, "Activating and Relaxing Music Entrain the Speed of Beat Synchronized Walking," *PLoS ONE*, vol. 8, no. 7, Jul. 2013.
- [9] G. E. Yoo and S. J. Kim, "Rhythmic Auditory Cueing in Motor Rehabilitation for Stroke Patients: Systematic Review and Meta-Analysis," *Journal of Music Therapy*, vol. 53, no. 2, pp. 149–177, 04 2016.
- [10] C. Nombela, L. E. Hughes, A. M. Owen, and J. A. Grahn, "Into the groove: Can rhythm influence Parkinson's disease?" *Neuroscience & Biobehavioral Reviews*, vol. 37, no. 10, pp. 2564–2570, Dec. 2013.
- [11] C. P. Hurt, R. R. Rice, G. McIntosh, and M. Thaut, "Rhythmic auditory stimulation in gait training for patients with traumatic brain injury," *Journal of music therapy*, vol. 35, pp. 228–241, 02 1998.
- [12] N. Schaffert, T. B. Janzen, K. Mattes, and M. H. Thaut, "A review on the relationship between sound and movement in sports and rehabilitation," *Frontiers in Psychology*, vol. 10, p. 244, 2019.
- [13] M. H. Thaut and M. Abiru, "Rhythmic Auditory Stimulation in Rehabilitation of Movement Disorders: A Review Of Current Research," *Music Perception: An Interdisciplinary Journal*, vol. 27, no. 4, pp. 263–269, 2010.
- [14] P.-J. Maes, J. Buhmann, and M. Leman, "3mo: A model for music-based biofeedback," *Frontiers in neuroscience*, vol. 10, p. 548, 2016.
- [15] G. Kramer, B. Walker, T. Bonebright, P. Cook, J. Flowers, N. Miner, and Neuhoff. (1999) Sonification report: Status of the Field and Research Agenda.
- [16] D. Zanotto, G. Rosati, S. Spagnol, P. Stegall, and S. K. Agrawal, "Effects of complementary auditory feedback in robot-assisted lower extremity motor adaptation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, pp. 775–786, 2013.
- [17] J. Perry, S. T. k, and J. R. Davids, "Gait Analysis: Normal and Pathological Function," *Journal of Pediatric Orthopaedics*, vol. 12, no. 6, 1992.
- [18] R. Enoka, *Neuromechanical Basis of Kinesiology*. Human Kinetics Books, 1988.
- [19] S. Lord, B. Galna, and L. Rochester, "Moving forward on gait measurement: Toward a more refined approach," *Movement Disorders*, vol. 28, no. 11, pp. 1534–1543, 2013.
- [20] "IM Gait Mate." [Online]. Available: <http://www.interactivemetronome.com/IMW/IMPUBLIC/products.aspx>
- [21] B. Moens, C. Muller, L. van Noorden, M. Franěk, B. Celie, J. Boone, J. Bourgois, and M. Leman, "Encouraging spontaneous synchronisation with d-jogger, an adaptive music player that aligns movement and music," *PloS one*, vol. 9, no. 12, p. e114234, 2014.
- [22] B. Moens, "D-jogger : An interactive music system for gait synchronisation with applications for sports and rehabilitation," Ph.D. dissertation, Ghent University, 2018.
- [23] Delsys™, "4-channel FSR Adapter User's Guide," pp. 10–18.
- [24] C. L. Vaughan, Ed., *Dynamics of Human Gait*, 2nd ed. Kiboho Publishers, 1999.
- [25] S. Lord, B. Galna, J. Verghese, S. Coleman, D. Burn, and L. Rochester, "Independent domains of gait in older adults and associated motor and nonmotor attributes: Validation of a factor analysis approach," *The Journals of Gerontology. Series A, Biological Sciences and Medical Sciences*, 2012.

# FROM VOCAL SKETCHING TO SOUND MODELS BY MEANS OF A SOUND-BASED MUSICAL TRANSCRIPTION SYSTEM

Claudio Panariello, Mattias Sköld<sup>†</sup>, Emma Frid, Roberto Bresin

Sound and Music Computing

KTH Royal Institute of Technology

<sup>†</sup>KMH Royal College of Music

{claudiop,maskold,emmafrid,roberto}@kth.se

## ABSTRACT

This paper explores how notation developed for the representation of sound-based musical structures could be used for the transcription of vocal sketches representing expressive robot movements. A mime actor initially produced expressive movements which were translated to a humanoid robot. The same actor was then asked to illustrate these movements using vocal sketching. The vocal sketches were transcribed by two composers using sound-based notation. The same composers later synthesized new sonic sketches from the annotated data. Different transcriptions and synthesized versions of these were compared in order to investigate how the audible outcome changes for different transcriptions and synthesis routines. This method provides a palette of sound models suitable for the sonification of expressive body movements.

## 1. INTRODUCTION

In this paper we present work conducted within the scope of the SONAO project, introduced in [1]. SONAO aims to improve the comprehensibility of robot non-verbal communication (NVC) through an increased clarity of robot expressive gestures and non-verbal sounds. The purpose of the SONAO project is to incorporate movement sonification in Human Robot Interaction (HRI), i.e. to use movement sonification to produce expressive sounds. Up to this point, movement sonification has only been used to a very limited extent in social robotics (see e.g. [2, 3]). Despite the fact that sounds produced by robots can affect the interaction with humans, sound design is often an overlooked aspect in HRI. Although some research has focused on developing sounds for humanoid robots such as NAO<sup>1</sup> (see e.g. [4–6]), sounds used in HRI have traditionally been based on rather simple synthesis methods, or on pre-recorded samples. Design decisions as well as mapping strategies are rarely described and motivated in these contexts. Moreover, those who design the robot sounds often lack musical training.

<sup>1</sup> <https://www.softbankrobotics.com/emea/en/nao>

Copyright: © 2019 Claudio Panariello et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

In the study presented in this paper, a framework for sound design in HRI is proposed, based on a work-flow starting from recordings of expressive gestures performed by a mime-actor, translated into non-linguistic sounds through vocal sketches, which in turn are annotated using a music annotation system. By incorporating composers in the design process, we hope to gain insight into how vocalizations could be used as a design material in the context of Human Robot Interaction (HRI), through translations into abstract musical representations.

## 2. BACKGROUND

The current study makes use of vocal sketching as a prototyping tool for exploration of sound design in HRI. Vocal sketching involves the use of the voice and body to demonstrate the relationship between actions and sonic feedback [7] and has successfully been used in a wide range of different projects, for example in SkAT-VG [8].

The notation system used for transcription in this study is part of an ongoing research project at KTH Royal Institute of Technology and KMH Royal College of Music, exploring the possibilities of representing pitch-based and sound-based music for composition [9, 10]. By using notation that combines the possibilities of electroacoustic music analysis with traditional music notation, we can describe sound structures with great detail. The notation symbols were adapted from concepts and symbols by Thoresen and Hedman [11], whose notation system for music analysis combines Pierre Schaeffer’s ideas on sound classification [12] with Denis Smalley’s theories of spectromorphology [13]. Placing symbols, aimed for phenomenological analysis, over a fixed time-frequency grid enables the transcription and re-synthesis of sound structures. The notation system presented in [9, 10] had previously been successfully tested with several students at KMH Royal College of Music, where findings suggested that different composers could synthesize very similar sonic results starting from same notation.

Up to this point, there has been relatively little research on how musical transcription could be used in the context of sonification. In particular, few attempts have aimed to merge the fields of electronic music with HRI. In seminal work by choreographer Åsa Unander-Scharin, expressive robot movements have been used for choreographing contemporary versions of classical music compositions by

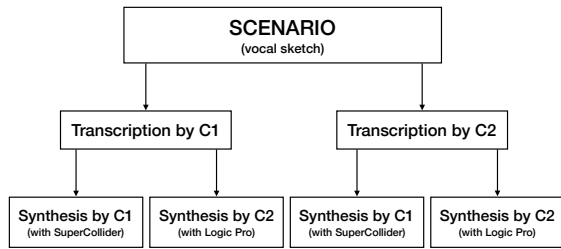


Figure 1. Flow chart of the transcription and synthesis process for composers C1 and C2, for each of the three scenarios (vocal sketches).

Monteverdi<sup>2</sup> and Tchaikovsky [14].

### 3. METHOD

#### 3.1 Procedure

The current paper emanates from material presented in the dataset described in [15]. This dataset consists of videos, motion capture data and audio recordings of a mime-actor portraying five inner states and emotions. A subset of these videos was used in a workshop with the same mime-actor, in which he vocalized sounds associated with respective emotion (and corresponding expressive gesture). Videos of the mime actor performing the three gestures used in this study are available online<sup>3</sup>. An example of the mime-actor performing one of the gestures is displayed in Fig. 3. We also interviewed the mime actor about which parts of the body that were essential in the communication of the emotions through respective gestures. In the current study, a selection of recordings from this vocal sketching session was used as basis for a composition task. Vocalizations expressing the following emotions were opted for: frustrated, relaxed and sad.

Two composers, author 1 (C1) and author 2 (C2), listened to the vocal sketches and transcribed them using the notation system described in section 2. Each composer worked on the transcription independently, resulting in a total of two transcriptions per scenario. Then, all the transcriptions were used by both composers as a starting point for synthesis of new sonic sketches. Every composer produced two different sonic sketches per scenario, one for each transcription. This methodology was used to ensure that the composers did not only re-synthesize their *own* transcriptions. In the end, the number of sonic sketches was four for respective scenario, giving us a total of 12 sonic sketches. This process is outlined in Fig. 1. The final synthesized sketches were then compared in order to investigate how they were affected by the transcription and the different synthesis routine adopted by the two composers.

#### 3.2 Material

Three of the vocalizations performed in the vocal sketching experiment described above were used in the current study: one vocalization of a frustrated gesture (called "Scenario

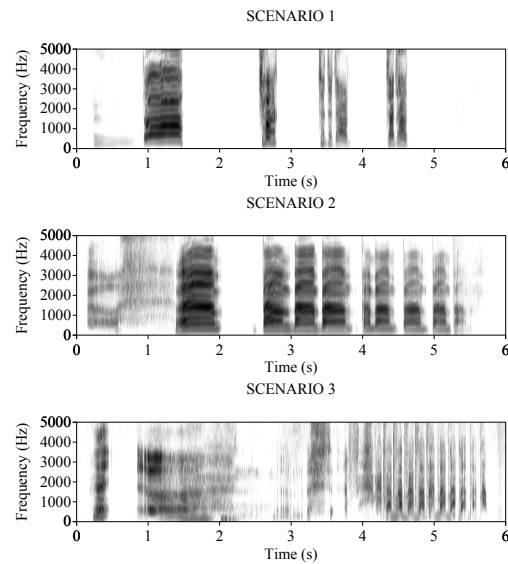


Figure 2. Spectrograms of the vocalizations for scenario 1-3.



Figure 3. Mime-actor performing a "frustrated" gesture.

1"), one vocalization of a relaxed gesture (called "Scenario 2"), and one vocalization of a gesture going from sad to reassuring (called "Scenario 3"). The three scenarios included the following dialogues:

**Scenario 1** Actor: "Everyone can you please line up to the left."

**Scenario 2** Actor: "Everyone can you please line up to the left."

**Scenario 3** Actor: "Sorry I broke this glass."  
Interlocutor: "No problem, I'll fix it."

The same phrase was used for Scenario 1 and 2, however, the level of emotional expression was different for the two. Sound files are available online<sup>4</sup>. Spectrograms of respective vocal sketch are shown in Fig. 2.

### 4. RESULTS

#### 4.1 Transcriptions

The two composers transcribed all three vocal sketches independently, resulting in a total of six scores. Comparing the two transcriptions for each scenario, we could observe that the transcriptions were similar in terms of the overall gestures, rhythm and pitch. Fig. 4 shows the two

<sup>2</sup> <http://www.operamecatronica.com/node/1171>

<sup>3</sup> <https://kth.box.com/v/robotsonification>

<sup>4</sup> <https://kth.box.com/v/robotsonification>

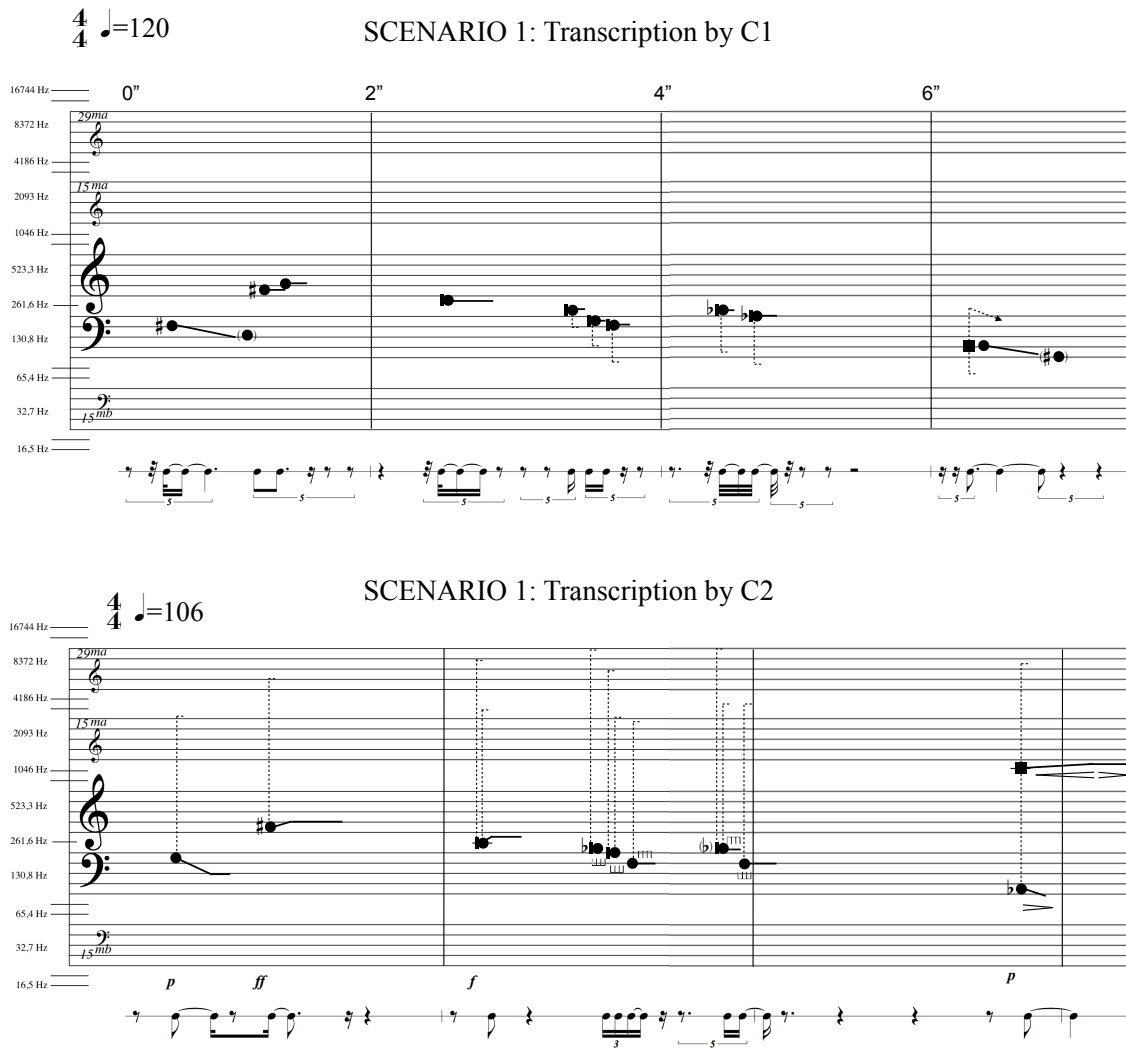


Figure 4. Transcriptions of Scenario 1 by the two composers.

analyses of Scenario 1: both composers agreed in notating the initial glissando pitched sound followed by another short glissando in the middle register and then in notating five pitched sounds with a complex onset; the two analysis both end with a short glissando in the low register overlapped with a complex sound. However, there are some differences in the notation of the spectral width. These discrepancies with regard to timbre were to be expected since notating the spectral content of a sound over a fixed time-frequency grid is not a standardized method of analysis for composers, even in the field of electroacoustic music. In so saying, the notation method leads to some approximations in the graphical representation that affects the synthesis. As a matter of fact, the two transcription of Scenario 3 are the ones that show the most significant differences, as can be seen in Fig. 5: the original vocal sketch was indeed composed by a high number of non-pitched throat sounds with a complex timbre, which are hard to notate in an univocal way. As can be noticed, there were different notation solutions for the vocal sketches' more growling sounds, where

C1 choose to notate them as inharmonic sounds with diamond noteheads, while C2 used the comb-like symbol that signifies a granular energy articulation. Nevertheless, the two composers agreed on the rhythmic transcription and on the general trend of the sonic structure of Scenario 3 (starting in the middle register, going to the low, then raising to the high register and ending with an iterated sound in the low register again).

## 4.2 Sound Synthesis

Both composers realized sound synthesis from all six scores, resulting in a total of 12 synthesized sound files. Composer C1 realised them using only the SuperCollider programming environment<sup>5</sup>; composer C2 used only Logic Pro<sup>6</sup>.

<sup>5</sup><https://supercollider.github.io/>

<sup>6</sup><https://www.apple.com/logic-pro/>



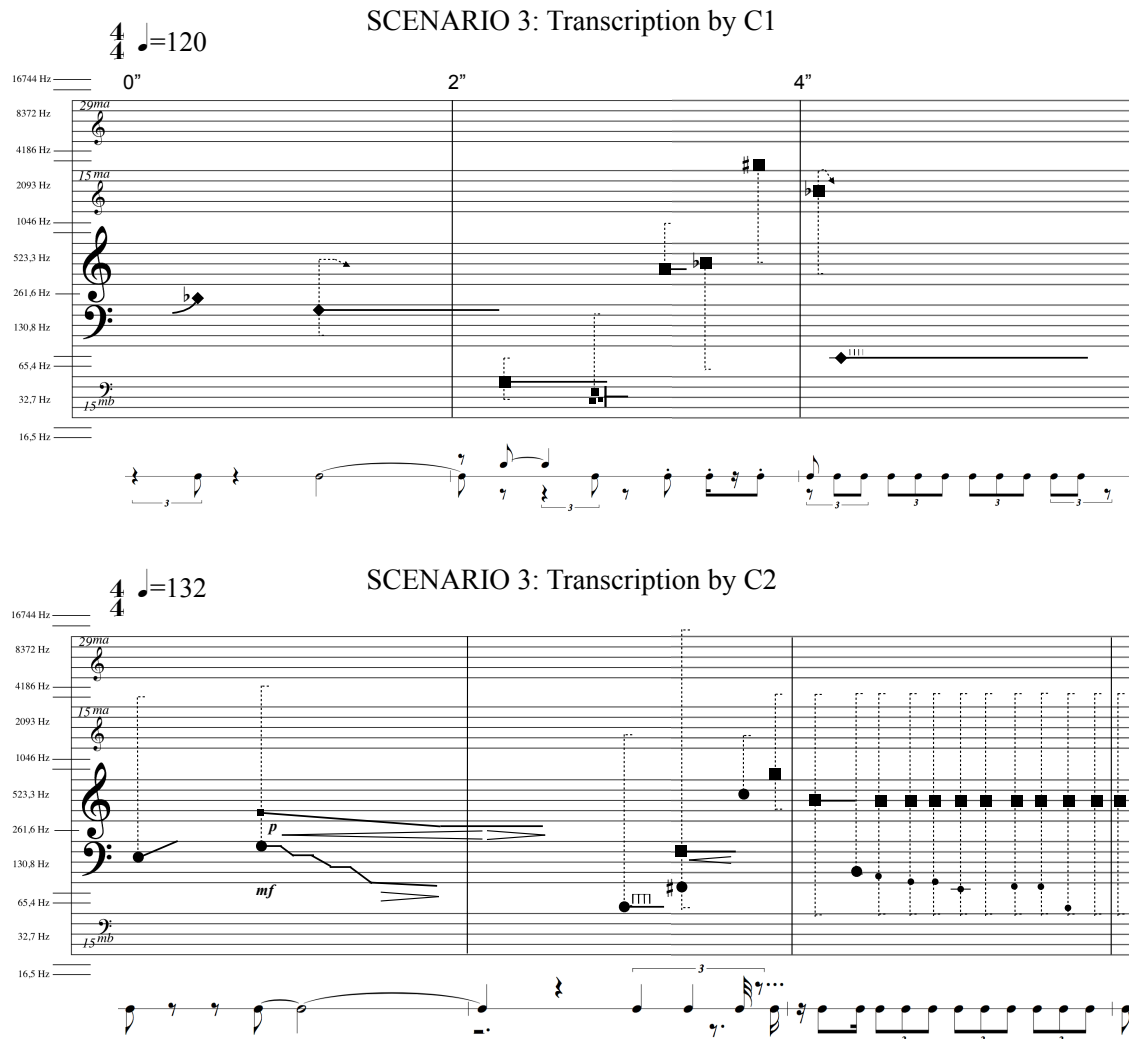


Figure 5. Transcriptions of Scenario 3 by the two composers.

#### 4.2.1 Synthesis with SuperCollider

For the synthesis in SuperCollider, three main Synths were built in order to recreate the three main categories used in the analysis. The noise-based sound was created using different instances of subtractive synthesis; the pitched sound was realized using a filtered sawtooth; the pitched sound with inharmonic spectrum was designed using an inharmonic additive synthesis of filtered noise generators.

All the Synths had the possibility to be shaped with a parametric envelope and to be granularized using the GrainIn unit generator. The output of each Synth was sent into a reverberation module.

The score was then created on the client side using the Task, that is a pauseable process. Two arrays were initialized with the durations and the main pitches found in the analysis step, and they were used to schedule all the sound events. For each of them, one or more Synths were initialized with all the appropriate parameters. This process is

summarized in the code presented in Listing 1.

The SuperCollider patches are available online <sup>7</sup>.

Listing 1. SuperCollider patch structure.

```
//Definition of Synths
SynthDef(\noise, {...}).send(s);
SynthDef(\pitch, {...}).send(s);
SynthDef(\dytonic, {...}).send(s);
SynthDef(\rev, {...}).send(s);
//
(
  //Score
  ~durations = [...];
  ~pitch = [...];
  t = Task({
    ... //Sound events//...
  }).start;
)
```

<sup>7</sup> <https://kth.box.com/v/robotsonification>

#### 4.2.2 Synthesis with Logic Pro

For the synthesis in Logic Pro, three instances of the ES2 virtual analog synthesizer plugin were used. The layout of the ES3 is similar to that of the Minimooog, but with some digital advantages such as 100 single-cycle waveforms for the oscillators. The sound objects of the notation were synthesized using combinations of filtered sawtooth oscillators and noise generators. For articulation and dynamics, the ES2 volume envelope was used for short durations and Logic's track volume automation was used for longer durations. For more flexible control and also automation of spectral width, separate channel EQs with low-pass and high-pass filters were added, mainly for instances playing the non-pitched noise-based sounds. Iteration and granularity were generated using LFOs controlling amplitude modulation in the ES2 modulation matrix.

#### 4.2.3 Results

Despite the differences in choices of sound synthesis software, the produced sound files showed great similarities. Many of the vocal sketch sounds were either pitched or complex (non-pitched) sounds, which for both sets of the synthesized scores translated into filtered saw-tooth waves and filtered noise. Fig. 6 shows the original vocal sketch compared to the two sound synthesis of Scenario 1 made by C1 and C2. Moreover, there is also great compatibility between these sound synthesis and the ones the composers realized from the transcription of the other: C1's synthesis of C2's transcription is really similar to C1's synthesis of his own transcription, and vice-versa. This shows that, starting from the same transcription, the different synthesized versions sound the same, proving the effectiveness of the notation system.

Similar results could be observed for Scenario 2: the transcriptions were similar and there were no doubts in identifying the sound events as complex or pitched. The sound synthesis results were very similar as well.

Interestingly, the case of Scenario 3 was a bit different from the prior scenarios. The original vocal sketch was harder to notate in regards to the spectral content. The two transcriptions lead to synthesized sounds that barely resemble the original vocal sketch. Despite this, when the composers synthesized over the other's transcription, the results are again compatible with the previous synthesis, as expected.

## 5. DISCUSSION

The challenge in transcribing non-musical sounds is similar to that of analysing electroacoustic music. One must decide what parameters to account for and with what level of detail. Clearly audible onsets of purely pitched or noisy sounds are easier to describe (as shown in the cases of Scenario 1 and Scenario 2) than intricate combinations of sound where elements of pitch and noise are intertwined and transformed over time (case of Scenario 3). Still the "musical identity" of the vocal sketches remained intact as they were translated into scores and back into synthesized sound. This was also noted when the same notation system

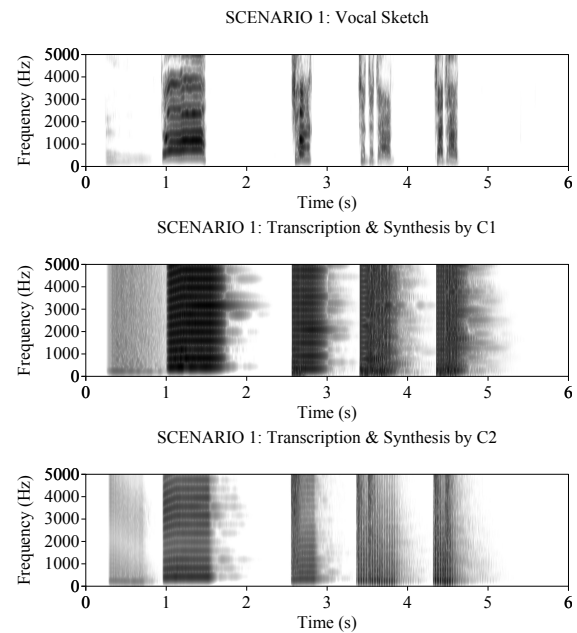


Figure 6. Spectrograms of the vocalization for Scenario 1, and synthesized versions by the two composers from their own transcriptions.

was used for the interpretation of musical structures [10]. Indeed the synthesized versions of each scenario made by the two composers were judged to be perceptually very similar in informal listening tests made by expert listeners at KTH. Still, there are discrepancies between the vocal sketches and the synthesized versions. There are two ways of dealing with them: one is to aim for transcriptions with much greater detail in an attempt to capture the voice more fully, the other is to think of the notation's function as the preserver of a sound structure's basic identity and consider some features of the vocal sketch the interpretations of the sound structure itself.

The new method presented in this paper, based on the sonic rendering of transcriptions of vocal sketches of body movements, provides a palette of sound models suitable for the sonification of expressive body movements. In particular, in the framework of the SONAO project [1] we are interested in identifying a set of sound models which can be used as a starting point for the design of real-time sonic representation of humanoid-robots expressive movements.

## 6. CONCLUSIONS

We have showed how notation developed for sound-based musical structures can be used for representing vocal sketches depicting robot movements. Traditional music notation will typically capture the fundamental sound structure of the music, leaving interpretation and emotional expression to the performer. Similarly, what constitutes a sad vocal sound structure will not necessarily translate into a sad synthesized version of its score. This depends on what vocal features that were used to convey the feeling and with what level of detail the sound passage

was notated. However, using notated sound structures as blueprints for sonified movements is conceptually different from other forms of sonification in that the movements are not directly sonified, but connected to notated structures in the form of a score to be interpreted. This way of working opens a space for the sonic interpretation of the movements where certain structural relations between specific movements and their sounding counterparts remain the same while other features are interpreted depending on the context.

## 7. FUTURE WORK

The study presented in this paper will be followed by formal and extensive listening experiments focusing on the perceptual distance between vocal sketches and their synthesis. Some possible applications and future work are described below.

### 7.1 Sonification of Robot Gestures

During the interview with the mime-actor, he emphasized that the following parts of the body were important in the communication of the sad gesture in Scenario 3 were the hands, and possibly also the shoulders. For the frustrated and relaxed gestures in Scenario 1 and 2, he also mentioned that the hands should be emphasized. This connection between the body movement and the vocal sketch will be used in a future stage of the project: having all the MoCap data of the mime gestures, it will be possible to use them to control the sound synthesis, i.e. sonification, focusing on the parts of the body that was indicated by the mime actor himself as being the most important ones.

### 7.2 The Notation of Movement and Sound

Expanding on the possibilities of notation with regard to a robot's expressive movements and sounds, there is the possibility of also notating the movements, placing both gesture and sound on the same conceptual level. There is a rich tradition of notating both movement and sound in dance, and notation systems like Labanotation [16], often used for notating dance movements, have already been used in the design of movement-based interaction [17] and in interactive dance performances (see for example recent works by Daniel Zea<sup>8</sup>).

### Acknowledgments

The authors would like to thank Simon Alexanderson and Alejandro Bonnet for their valuable contributions to the SONAO project. We also thank the three anonymous reviewers for very helpful comments that contributed to improve the quality of our paper. This project was funded by Grant 2017-03979 from the Swedish Research Council and by NordForsk's Nordic University Hub "Nordic Sound and Music Computing Network - NordicSMC", project number 86892.

<sup>8</sup><http://danielzea.org/works/>

## 8. REFERENCES

- [1] E. Frid, R. Bresin, and S. Alexanderson, "Perception of Mechanical Sounds Inherent to Expressive Gestures of a NAO Robot-Implications for Movement Sonification of Humanoids," in *Proceedings of the Sound and Music Computing Conference*, 2018.
- [2] R. Zhang, M. Jeon, C. H. Park, and A. Howard, "Robotic sonification for promoting emotional and social interactions of children with ASD," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*. ACM, 2015, pp. 111–112.
- [3] J. Bellona, L. Bai, L. Dahl, and A. LaViers, "Empirically Informed Sound Synthesis Application for Enhancing the Perception of Expressive Robotic Movement," in *Proceedings of the International Conference on Auditory Display*. Georgia Institute of Technology, 2017.
- [4] J. Monceaux, J. Becker, C. Boudier, and A. Mazel, "First Steps in Emotional Expression of the Humanoid Robot NAO," in *Proceedings of the 2009 International Conference on Multimodal Interfaces*. ACM, 2009, pp. 235–236.
- [5] M. Häring, N. Bee, and E. André, "Creation and Evaluation of Emotion Expression with Body Movement, Sound and Eye Color for Humanoid Robots," in *Roman, 2011 Ieee*. IEEE, 2011, pp. 204–209.
- [6] A. Kappas, D. Küster, P. Dente, and C. Basedow, "Simply the BEST! Creation and Validation of the Bremen Emotional Sounds Toolkit," in *International Convention of Psychological Science*, 2015.
- [7] D. Rocchesso, S. Serafin, and M. Rinott, "Pedagogical approaches and methods," *Sonic Interaction Design*, pp. 125–150, 2013.
- [8] D. Rocchesso, G. Lemaitre, P. Susini, S. Ternström, and P. Boussard, "Sketching sound with voice and gesture," *interactions*, vol. 22, no. 1, pp. 38–41, 2015.
- [9] M. Sköld, "The Harmony of Noise: Constructing a Unified System for Representation of Pitch, Noise and Spatialization," in *CMMR2017 13th International Symposium on Computer Music Multidisciplinary Research*. Les éditions de PRISM, 2017, pp. 550–555.
- [10] Sköld, M., "Combining Sound- and Pitch-Based Notation for Teaching and Composition," in *TENOR'18 – Fourth International Conference on Technologies for Music Notation and Representation*, 2018, pp. 1–6.
- [11] L. Thoresen and A. Hedman, "Spectromorphological Analysis of Sound Objects: An Adaptation of Pierre Schaeffer's Typomorphology," *Organised Sound*, vol. 12, no. 2, pp. 129–141, 2007.
- [12] P. Schaeffer, *Treatise on Musical Objects: An Essay Across Disciplines*. Univ of California Press, 2017, vol. 20.

- [13] D. Smalley, "Spectromorphology: Explaining Sound-Shapes," *Organised Sound*, vol. 2, no. 2, pp. 107–126, 1997.
- [14] Å. Unander-Scharin, "Activity: La Robot-Cygne : Choreographic Reflections on Dancing Through a Mechatronical Double," 2009, startdatum: 24/09/2009; Slutdatum: 24/09/2009; Roll: Föreläsare; Typ: Föreläsning / muntligt bidrag.
- [15] S. Alexanderson, C. O'sullivan, M. Neff, and J. Beskow, "Mimebot — Investigating the Expressibility of Non-Verbal Communication Across Agent Embodiments," *ACM Transactions on Applied Perception (TAP)*, vol. 14, no. 4, p. 24, 2017.
- [16] A. H. Guest, *Labanotation: the System of Analyzing and Recording Movement*. Routledge, 2013.
- [17] L. Loke, A. T. Larssen, and T. Robertson, "Labanotation for Design of Movement-Based Interaction," in *Proceedings of the second Australasian conference on Interactive entertainment*. Creativity & Cognition Studios Press, 2005, pp. 113–120.



# TEMPO AND METRICAL ANALYSIS BY TRACKING MULTIPLE METRICAL LEVELS USING AUTOCORRELATION

**Olivier Lartillot**

RITMO Centre for Interdisciplinary Studies  
in Rhythm, Time and Motion  
University of Oslo  
olivier.lartillot@imv.uio.no

**Didier Grandjean**

Department of Psychology  
Swiss Center for Affective Sciences  
University of Geneva  
Didier.Grandjean@unige.ch

## ABSTRACT

We present a method for tempo estimation from audio recordings based on signal processing and peak tracking, and not depending on training on ground-truth data. First an accentuation curve, emphasising the temporal location and accentuation of notes, is based on a detection of bursts of energy localised in time and frequency. This enables to detect notes in dense polyphonic texture, while ignoring spectral fluctuation produced by vibrato and tremolo. Periodicities in the accentuation curve are detected using an improved version of autocorrelation function. Hierarchical metrical structures, composed of a large set of periodicities in pairwise harmonic relationships, are tracked over time. In this way, the metrical structure can be tracked even if the rhythmic emphasis switches from one metrical level to another.

This approach, compared to all the other participants to the MIREX Audio Tempo Extraction from 2006 to 2018, is the third best one among those that can track tempo variations. While the two best methods are based on machine learning, our method suggests a way to track tempo founded on signal processing and heuristics-based peak tracking. Besides, the approach offers for the first time a detailed representation of the dynamic evolution of the metrical structure. The method is integrated into *MIRtoolbox*, a Matlab toolbox freely available.

## 1. INTRODUCTION

Detecting tempo in music and tracking the evolution of tempo over time is a topic of research in MIR that has been extensively studied these last decades. Recently approaches based on deep learning have contributed to an important progress in the state of the art [1, 2]. In this paper, we present a method that relates to a more classical approach based on signal processing and heuristics-based data extraction. We previously briefly presented the principles of the approach [3]. This paper offers a more detailed description of the method.

One particularity of the proposed approach is that it enables to track not only one, two or three, but a larger number of metrical levels. This enables to get a detailed description of the dynamic evolution of the metrical structure: not only how the whole structure speeds up or slows down with respect to global tempo, but also how individual metrical levels might be emphasised at particular moments in the music. In order to give an indication of metrical activity that would not reduce solely on tempo but takes into consideration the activity on the various metrical levels, we introduce a new measure, called *dynamic metrical centroid*.

## 2. RELATED WORK

### 2.1 Accentuation curve

The estimation of tempo starts from a temporal description of the location and strength of events appearing in the piece of music. This first step consists in inferring an “onset detection curve”, also called *accentuation curve* [4]. Musical events are indicated by peaks; the height of each peak is related to the importance of the related event. *Envelope*-based approach globally estimates the energy for each successive temporal frame without considering its spectral decomposition; *spectral flux* methods estimate the difference of energy over successive frames on individual frequencies individually, and further summed together [5, 6]. The envelope approach would work in the case of sequences made of notes sufficiently isolated or accentuated with respect to the background, corresponding to short bursts of energy separated by low-energy transitions, as in simple percussive sequences. Indeed, in such case, the resulting envelope curve would show each percussive event with a peak. On the contrary, for dense musical sequences featuring overlapped notes, such as complex orchestral sounds, the spectral flux method better distinguishes the attack of individual notes, provided that the different notes occupy distinct frequency bands. Minor energy fluctuation along particular frequencies may blur the resulting accentuation curve in the point of making it impossible to detect the actual note attacks. The use of thresholding can filter out energy fluctuation on constant frequency bands (such as *tremolo*) and select only significantly high energy bursts related to note attacks. Still, energy fluctuating in frequency, such as *vibrato*, may still add noise to the resulting accentuation curve.

Copyright: © 2019 Olivier Lartillot et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

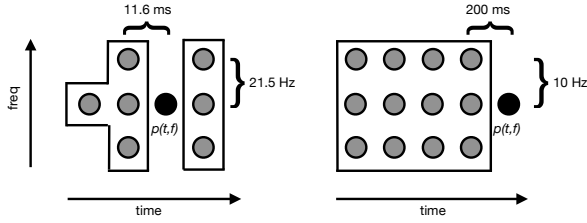


Figure 1. Comparison between the method in [7] (left) and our proposed method (right), for the comparison of a given spectrogram amplitude  $p(t, f)$  at time  $t$  and frequency  $f$  with amplitudes from previous (and next) frames.

In order to detect significant energy bursts on highly localised frequency ranges but still filter out the artifacts due to the possible frequency fluctuation along time of such localised events, it is necessary to add some tracking capability. The approach presented by [7] can be considered as an answer to this problem. It searches for rapid increase of amplitude on particular frequency components, and evaluates for each detected onset its “degree of onset”, defined as the rapidity of increase in amplitude. To estimate this increase of amplitude at a given time  $t$  for a particular frequency  $f$ , the amplitude  $p(t, f)$  is compared not only to the corresponding amplitude at the previous frame  $p(t-1, f)$ , but also to the amplitude at the higher and lower frequency bins  $p(t-1, f-1)$  and  $p(t-1, f+1)$  as well as  $p(t-2, f)$ . These previous points form the *contextual background*. The current amplitude  $p(t, f)$  is compared with the maximum of the amplitude at those four previous points, as shown in Figure 1. Let  $pp(t, f)$  be this maximum. Besides, the corresponding amplitudes at frame  $t+1$  are also compared with  $pp(t, f)$ .

For a given instant  $t$  and frequency  $f$ , the degree of onset is given by

$$do(t, f) = p(t, f) - pp(t, f) \quad (1)$$

The degrees of onset are summed over the frequency components, leading to the onset curve.

## 2.2 Periodicity analysis

A pulsation corresponds to a periodicity in the succession of peaks in the accentuation curve. Classical signal-processing methods estimate periodicity using methods such as autocorrelation, the YIN method, bank of comb-filter resonators with a constant half-time [8] or phase-locking resonators.<sup>1</sup> Basically, a range of possible periodicity frequencies is considered, and for each frequency, it is estimated whether there exists any periodicity at that frequency. In the following, we will call *periodicity function* the representation, such as autocorrelation function, showing the periodicity *score* related to each possible *period* (or alternatively each possible frequency).

## 2.3 Metrical structure

One common approach to extract the tempo from the periodicity function is to select the highest peak, within a range

<sup>1</sup> Cf. [4] for a detailed literature review.

of beat periodicities considered as most adequate, typically between 40 and 200 BPM, with a weighted emphasis on best perceived periodicity range. This approach fails when tracking the temporal evolution of tempo over time, especially for pieces of music where different metrical levels are emphasised throughout the temporal development. For instance, if at a given moment of the piece of music, there is an accentuated quarter note pulsation followed by an accentuated eighth note pulsation, the tempo tracking will switch from one BPM value to another one twice faster, although the actual tempo might remain constant. And as we may imagine, such shift from metrical level to another is very frequent in music.

In [4], three particular metrical levels are considered as core elements of the metrical structure: The *tactus* is considered as the most prominent level, also referred as the foot tapping rate or the *beat*. The tempo is often identified to the *tactus* level. The *tatum*—for “temporal atom”—is considered as the fastest subdivision of the metrical hierarchy, such that all other metrical levels (in particular *tactus* and *bar*) are multiples of that *tatum*. The *bar-level* or other metrical levels related to change of chords, melodic or rhythmic patterns, etc. The tracking of tempo along time result from a tracking of these three metrical levels using a Hidden Markov Model (HMM).

The *tatum* is considered (and modeled) as the minimal subdivision such that each other metrical level is a multiple of that elementary level, but this canonic situation does not describe all metrical cases: for instance, binary and ternary subdivisions often coexist, as we will see in section 5.

## 2.4 Deep-learning approaches

Recent deep-learning approaches start from the computation of a spectrogram, eventually followed by a filtering that emphasises the contrast between successive frames, along each different frequency [1]. In [1], the successive frames of the spectrogram are then fed into a Bidirectional Long Short-Term Memory (BLSTM) Recurrent Neural Network (RNN). This network can be understood as performing both the detection of events based on local contrast and the detection of periodicity in the succession of events, along multiple metrical levels. This is followed by a Dynamic Bayesian Network that plays a similar role as the HMM, tracking the pulsation along two metrical levels (corresponding to beats and downbeats). In [2], the whole process consists in feeding the spectrogram to a convolutional neural network (CNN).

## 3. PROPOSED METHOD

The proposed method introduces improvements in the successive steps forming the traditional procedure for metrical analysis that were presented in sections 2.1, 2.2 and 2.3. Those improvements are as follows. A modification of the localised method for accentuation curve estimation enables to better emphasise note onsets in complex polyphony with vibrato and tremolo (section 3.1). Periodicity detection is performed using a modified version of autocorrelation function (section 3.2).

Besides, we introduce a new methodology for tracking the metrical structure along a large range of periodicity layers in parallel. The tracking of the metrical structure is carried out in two steps:

1. a tracking of the *metrical grid* featuring a large range of possible periodicities (section 3.3). Instead of considering a fix and small number of pre-defined metrical levels, we propose to track a larger range of periodicity layers in parallel.
2. a selection of core metrical levels, leading to a *metrical structure*, which enables the estimation of metre and tempo (section 3.4).

### 3.1 Accentuation curve

Our method for the inference of the accentuation curve follows the same general principle of the model introduced in [7], detecting and tracking the apparition of partials locally in the spectrogram, as explained in section 2.1. In our case, the spectrogram is computed for the frequency range below 5000 Hz and the energy is represented in the logarithmic scale in decibel.

We use different parameters for the specification of the temporal scope and the frequency width of the contextual background. In [7], the frequency width is of 43 Hz and the temporal depth of 23 ms. After testing on a range of musical styles, we chose a frequency width around 20 Hz and a temporal depth of .8 second (cf. Figure 1). By enlarging the temporal horizon of the contextual background, this enables to filter out *tremolo* effects and to focus on more prominent increase of energy.

In the proposed model, the second condition for onset detection specified in [7]—namely, that the energy on the frame succeeding the current one should be higher than the contextual background—is withdrawn, for the sake of simplicity. That constraint seems aimed at filtering out bursts of energy that are just one frame long, but bursts two frames long would not be filtered out. And we might hypothesise that short bursts of energy might still be perceived as events.

Finally, the degree of onset is different from the one proposed in [7]. Instead of conditioning the degree of onset to the increase of energy with respect to the contextual background, we propose instead to condition it to the absolute level of energy:

$$do(t, f) = p(t, f) \quad (2)$$

This is because a burst of energy of a given level  $p(t, f)$  might be perceived as strong, and could contribute therefore to the detection of a note onset, even if there was a relatively loud sound in the frequency and temporal vicinity. This modification globally improved the results in our tests.

In our proposed method, the accentuation curve shows more note onsets than in [7]. This leads to a more detailed analysis of periodicity and a richer metrical analysis. This allows sometimes the discovery of the underlying metrical structure that was hidden under a complex surface and was not detected using [7].

### 3.2 Periodicity analysis

Tempo is estimated by computing an autocorrelogram with a frame length of 5 seconds and hop factor 5%, for a range of time lags between 60 ms and 2.5 s, corresponding to a tempo range between 24 and 1000 BPM. The autocorrelation curve is normalized so that the autocorrelation at zero lag is identically 1.

A peak picking is applied to each frame of the autocorrelogram separately. The beginning and the end of the autocorrelation curves are not taken into consideration for peak picking as they do not correspond to actual local maxima. A given local maximum will be considered as a peak if its distance with the previous and successive local minima (if any) is higher than 5% of the total amplitude (i.e., the distance between the global maximum and minimum) of the autocorrelation function.

One important problem with autocorrelation functions is that a lag can be selected as prominent because it is found often in the signal although the lag is not repeated successively. We propose a simple solution based on the following property: For a given lag to be repeated at least twice, the periodicity score associated with twice the lag should have a high probability score as well. This heuristics can be implemented as a single post-processing operation applied to the autocorrelation function, removing all periodicity candidate for which there is no periodicity candidate at around twice its lag.

### 3.3 Tracking the metrical grid

#### 3.3.1 Principles

In the proposed approach, we track a large range of possible metrical levels. This is done in two successive steps:

- the construction of a detailed set of periodicities inherent to the metrical structure, leading to what we propose to call a metrical *grid*, where individual periodicities are called *layers*,
- the selection among those metrical layers of core metrical *levels*, whose periods are in multiplicity ratios. All other layers of the metrical grid are simple multiples or submultiples of those metrical levels. One metrical level is selected as the most prevalent, for the determination of tempo.

For each metrical layer  $i$ , its *tempo*  $T_i$  (meaning the tempo related to the metrical grid by tapping on that particular metrical layer) and period  $\tau_i$  are directly related to the tempo  $T_1$  and period  $\tau_1$  of the reference layer  $i = 1$ :

$$T_i = \frac{T_1}{i}, \tau_i = \tau_1 i \quad (3)$$

For instance, the tempo at metrical layer 2 is twice slower than the one at metrical layer 1. Although tempo can change over time, the tempo related to the different metrical periodicities conserve their multiplicity ratio, so that equation 3 remains valid in theory.

The tracking of the metrical grid over time requires a management of uncertainty and noisy data. Periodicity

lags measured in the autocorrelogram do not exactly comply with the theoretical lags given by equation 3. For that reason, each metrical layer  $i$  is described by both:

- theoretically, the temporal series of periods  $\tau_i(n)$  related to metrical layer  $i$  knowing the global tempo given by  $\tau_1(n)$ .
- practically, the temporal series of lags  $t_i(n)$  effectively measured at location of peaks in the autocorrelation function for each successive frame  $n$ .

In the graphical representation of the metrical structure, both actual and theoretical periods are shown: the temporal succession of the theoretical values at a given metrical layer is shown with a line of dots, whereas the actual periods are indicated with crosses that are connected to the theoretical dot with a vertical line. For instance in Figure 2, we see a superposition of metrical layers, each with a label indicated on the left side, starting from layer 0.25 up until layer 4, with also a layer 4.25 appearing around 30 second after the start of the excerpt.

### 3.3.2 Procedure

The theoretical periods are inferred based on the measured periods, as we will see in equation 13.

The integration of peak into the metrical grids is done in three steps, related to the extension of metrical layers already registered, the creation of new metrical layers and finally the initiation of new metrical grids.

For each successive time frame  $n$ , peaks in the periodicity function (as specified in section 3.2) are considered in decreasing order of periodicity score. This is motivated by the observation that strongest periodicities, corresponding generally to important metrical levels, tend to show a more stable dynamic evolution and are hence more reliable guides for the tracking of the metrical structure. Weaker autocorrelation peaks, on the contrary, may sometimes result from a mixture of underlying local periodicities, hence might tend to behave more erratically. For each frame, the strongest peaks first considered enable to get a first estimation of the tempo  $T_1(n)$  at that frame, which will be used as a reference when integrating the weaker periodicities.

Each peak, related to a period (or lag)  $t$  is tentatively mapped to one existing metrical layer  $i$ . We consider two ways to estimate the distance between current peak  $t$  and a given metrical layer  $i$ : either by comparing current peak lag  $t$  with the actual lag of the peak associated with this metrical layer  $i$  at previous frame  $n - 1$ :

$$d_1(t, i) = |t - t_i(n - 1)| \quad (4)$$

or by comparing current peak lag  $t$  with the theoretical lag at that metrical layer  $i$  knowing the global tempo:

$$d_2(t, i) = |t - \tau_i(n)| \quad (5)$$

For low lag values, small difference in time domain can still lead to importance difference in tempo domain. For that reason, an additional distance is considered, based on tempo ratio:

$$d_3(t, i) = \left| \log_2 \left( \frac{t}{\tau_i(n)} \right) \right| \quad (6)$$

The distance between current peak  $t$  and a given metrical layer  $i$  can be then considered as the minimum of the two distances on the time domain:

$$d(t, i) = \min(d_1(t, i), d_2(t, i)) \quad (7)$$

and the closest metrical layer  $i^*$  can be chosen as the one with minimal distance:

$$i^* = \arg \min_i d(t, i) \quad (8)$$

If this metrical period has already been assigned to a stronger peak in current frame  $n$ , this weaker peak  $t$  is discarded for any further analysis. In other cases, its integration to the metrical period  $i^*$  is carried out if it is close enough, both in time domain ( $d(t, i)$ ) and in tempo domain ( $d_3(t, i)$ ):

$$d(t, i) < \delta \text{ and } d_3(t, i) < \epsilon \quad (9)$$

In a second step, we check whether the periodicity peak triggers the addition of a new metrical layer in that metrical grid:

- For all the slower metrical layers  $i$ , we find those that have a theoretical period that is in integer ratio with the peak lag  $t$ :

$$\min \left( \frac{\tau_i(n)}{t} \bmod 1, 1 - \left( \frac{\tau_i(n)}{t} \bmod 1 \right) \right) < \epsilon \quad (10)$$

where  $\epsilon$  is set to .02 if no other stronger peak in the current time frame  $n$  has been identified with the metrical grid, and else to .1 in the other case.

If we find several of those slower periods in integer ratio, we select the fastest one, unless we find a slower one with a ratio defined in equation 10 that would be closer to 0.

- Similarly, for all the faster metrical layers,  $i$  we find those that have a theoretical pulse lag that is in integer ratio with the peak lag:

$$\min \left( \frac{t}{\tau_i(n)} \bmod 1, 1 - \left( \frac{t}{\tau_i(n)} \bmod 1 \right) \right) < \epsilon \quad (11)$$

- If we have found both a slower and a faster period, we select the one with stronger periodicity score.
- This metrical layer, of index  $i_R$ , will be used as reference onto which the new discovered metrical layer is based. The new metrical index  $i^*$  is defined as:

$$i^* = i_R * \left\lceil \frac{t}{\tau_{i_R}(n)} \right\rceil \quad (12)$$

Finally, if the strongest periodicity peak in the given time frame  $n$  is strong enough (with periodicity score above a certain threshold  $\theta$ ) and is not associated with any period of the metrical grid(s) currently active, a new metrical grid is created, with a single metrical period (with  $i = 1$ ) related to that peak.



All active metrical grids are tracked in parallel, by tentatively mapping the peaks of the periodicity curve on the periods of each grid.

A metrical grid stops being further extended whenever there is no peak in the given frame that can extend any of the dominant periods. Mechanisms have also been conceived to fuse multiple grids whenever it turns out that they belong to a single hierarchy.

The global tempo associated to the metrical grid is updated based on the actual lags measured along the different metrical periods in the current frame  $n$ . For each metrical period  $i$  and for the peak lag  $t_i$  associated to it, we obtain a particular estimation of the global lag (i.e., the lag at periodicity index 1), namely  $\frac{t_i}{i}$ . We can then obtain a global estimation of the global lag by averaging these tempo estimation at different periods, using as a weight the autocorrelation score  $s_i$  of those peaks:

$$\tau_1(n) = \frac{\sum_{i \in D} s_i \frac{t_i}{i}}{\sum_{i \in D} s_i} \quad (13)$$

Not all metrical periods are considered, because there can be a very large number of those, and many of the higher periods are only redundant information that tend to be unreliable. For that reason, a selection of the most important—or *dominant*—metrical periods is performed, corresponding to the set  $D$  in previous equation. Each time a new metrical grid is initiated, the first metrical period ( $i = 1$ ) is considered as dominant. Any other metrical period  $i$  becomes dominant whenever the last peak integrated is strong (i.e., with an autocorrelation score higher than a given threshold  $\theta$ ) and if the reference metrical period upon which layer  $i$  is based is also dominant.

The actual updating of the global tempo is somewhat more complex than the description given in the previous paragraph, because we consider the evolution of the tempo from the previous frame to the current frame, and limit the amplitude of the tempo change up to a certain threshold. This enables to add a certain kind of “inertia” to the model such that unrelated periodicities in the signal will not lead to sharp discontinuity in the tempo curves.

Values used for some parameters defined in this section:  $\delta = .07$ ,  $\epsilon = .2$ ,  $\theta = .15$ .

### 3.4 Metrical structure

The metrical grids constructed by the tracking method presented in the previous paragraph are so far made of a mere superposition of metrical periods. The ratio number associated with each metrical level should be considered relatively. For instance, the value 1 has no absolute meaning, it is arbitrarily given to the first level detected. Level 1.5 is 3 times slower than level .5. For each metrical grid, one or several of its metrical periods have been characterized as dominant because of their salience at particular instants of the temporal development of the metrical grid, and because such selection offers helpful guiding points throughout the temporal tracking of the metrical grid. Yet these selected dominant metrical periods simply highlight particular articulation of the surface and do not necessarily relate to the core metrical levels of the actual metrical structure.

A metrical structure is composed of a certain number of metrical *levels*: they are particular periods of the metrical grid that are multiple of each other. For instance, in a typical meter of time signature 4/4, the main metrical level is the quarter note, the upper levels are the half note and the whole note, the lower levels are the eighth note, the sixteenth note, and any other subdivision by 2 of these levels. In the same example, dotted half note (corresponding to three quarter notes) is related to one metrical period in the metrical grid, because it is explicitly represented in the autocorrelation function as a possible periodicity, but it is not considered as a metrical *level*.

In the graphical representations shown in Figures 2, 3 and 4, the metrical levels are shown in black while the other metrical layers are shown in gray.

The metrical structure offers core information about meter. In particular, tempo corresponds to beat periodicity at one particular metrical level. In a typical metre, the main metrical level could be used as the tempo reference. In our example, with a typical time signature 4/4, the tempo could be inferred by reporting the period at the metrical level corresponding to the quarter note. However, in practice, there can be ambiguity related to the actual metre, and especially related to the choice of the main metrical level.

For each metrical periodicity  $i$  can be associated a numerical score  $S_i$ , computed as a summation across frames of the related periodicity score  $s_{i,n}$  for each frame  $n$ . The metrical periodicities  $i$  are progressively considered in decreasing order of score  $S_i$  as potential metrical levels.

In a first attempt, we integrate all possible periodicities as long as they form a coherent metrical structure. The metrical structure is initially made of one single metrical level corresponding to the strongest periodicity. Each remaining metrical period  $P$ , from strongest to weakest, is progressively compared with the metrical levels of the metrical structure, in order to check that for each metrical level  $L$ ,  $P$  has a periodicity that is a multiple of  $L$ , or reversely. In such case,  $P$  is integrated into the metrical structure as a new metrical level.

This method may infer incorrect metrical structures in the presence of a strong accentuated metrical period that is not considered as a metrical level. This often happens in syncopated rhythm. For instance, a binary 4/4 metre with strong use of dotted quarter notes could lead to strongest periodicities at the eighth note (let's set this period to  $i = 1$ ), dotted quarter note ( $i = 3$ ) and whole note ( $i = 8$ ). One example is the rhythmical pattern 123-123-12, 123-123-12, etc. In such case, if the periodicities related to dotted quarter note ( $i = 3$ ) is stronger than the periodicities related to whole note ( $i = 8$ ), the first method would consider the metre to be ternary, of the form 6/8 for instance.

In order to solve the limitation of the first method, a more elaborate method constructs all possible metrical structures, with metrical levels taken from the series of metrical periods from the input metrical grid. To each metrical structure is associated a score obtained by summing the score related to each selected level. The metrical structure with highest score is finally selected. In our example, alternative metrical structures are constructed, both for ternary rhythm—

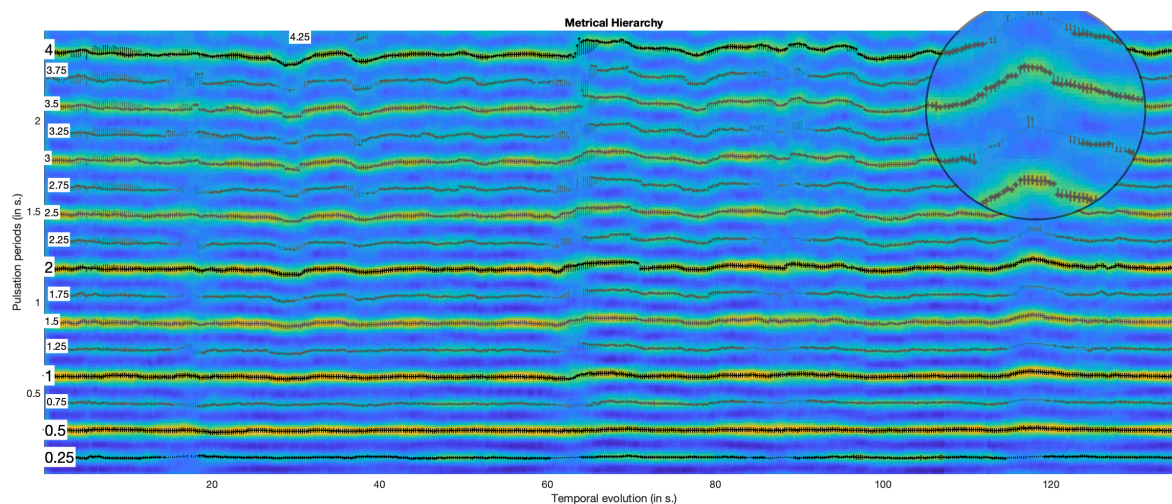


Figure 2. Autocorrelation-based periodogram with tracking of the metrical structure for the first 140 seconds of a performance of the first movement of J. S. Bach's *Brandenburg Concerto No. 2 in F major*, BWV 1047. Each metrical layer is indicated by a line of crosses extending from left to right, and preceded by a number indicating the index of the metrical layer. When the line is interrupted at particular temporal regions, the remaining dotted line represents the temporal tempo at that layer. Metrical levels are shown in black, while other metrical layers are shown in gray. See the text for further explanation.

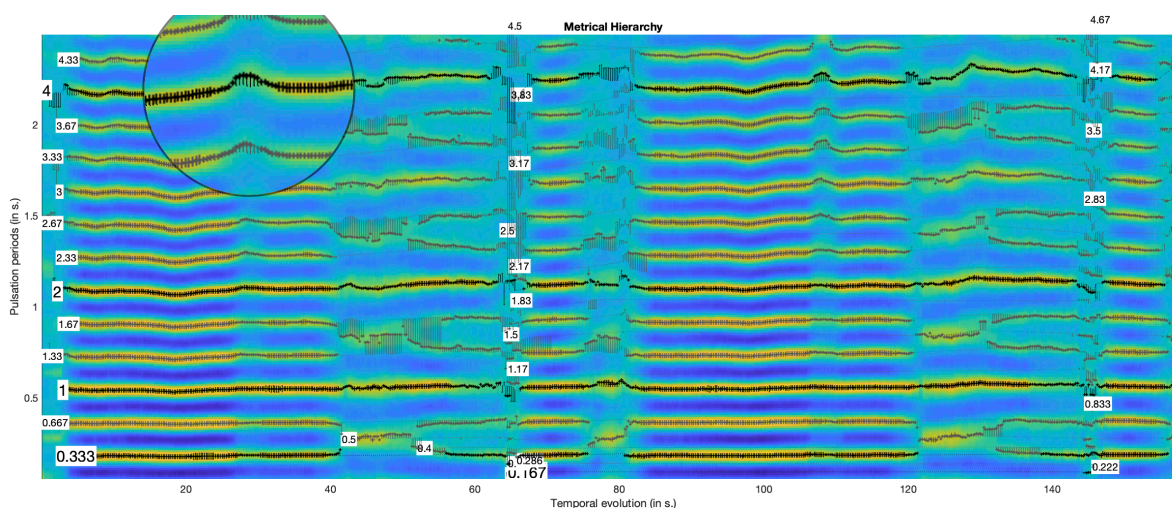


Figure 3. Autocorrelation-based periodogram with tracking of the metrical structure for the first 160 seconds of a performance of the Scherzo of L. van Beethoven's *Symphony No. 9 in D minor*, op.125, using the same graphical conventions as in Figure 2. As before, numbers, indicating metrical layer indices, are displayed where the metrical layers are first detected. For instance, layer 0.5, corresponding to the binary division of layer 1, appears at 40 seconds.

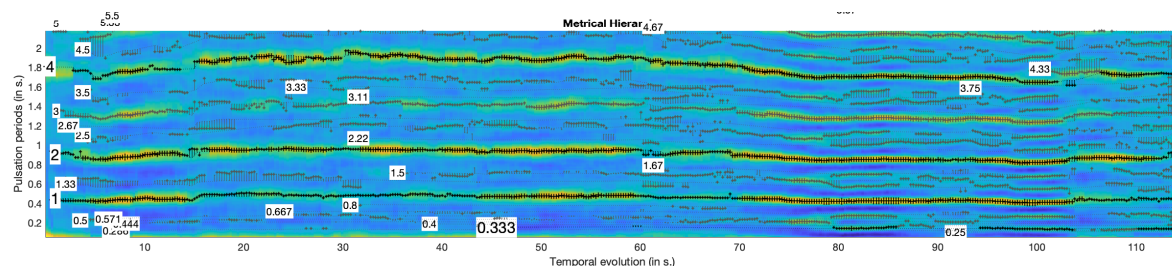


Figure 4. Autocorrelation-based periodogram with tracking of the metrical structure for the first 2 minutes of a performance of the *Allegro con fuoco* of A. Dvorak's *New World Symphony, Symphony No. 9 in E minor*, op. 95, B.178, using the same graphical conventions as in Figure 2.



with metrical levels (1, 3, 6), or (1, 3, 9), etc.—and for binary rhythm—(1, 2, 8), (1, 2, 4, 8), etc. If the periodicity corresponding to  $i = 8$  is sufficiently strong, the binary rhythm will be chosen by the model. Although  $i = 3$  is stronger than  $i = 8$ , the combination (1, 2, 8), for instance, can be stronger than the combination (1, 3, 6).

The resulting metrical structure is made of a combination of metrical levels, i.e., a subset  $(i_1, i_2, \dots)$  of the metrical periods of the metrical grid. One metrical level  $i_R$  needs to be selected as reference level for the computation of tempo. One simple strategy would consist in selecting the metrical level with highest score, as defined previously. However, those scores are based on purely signal processing method (namely, autocorrelation function), and do not take into account the fact that certain periodicities are more easily perceived than other. Studies have designed so-called “resonance curves” that enable to weight the periodicity score depending on the period, so that periods around typical range of periodicity around 120 BPMs would be preferred [9, 10]. We follow the same method, by weighting the metrical level scores  $S_{i_j}$  using the resonance curve proposed by [9], using as input to the resonance curve the median periodicity related to the given metrical level.

#### 4. COMPARATIVE EVALUATION

The original algorithm was submitted to the Audio Tempo Extraction competition under the MIREX (Music Information Retrieval Evaluation eXchange) annual campaign<sup>2</sup>. This evaluation is made using 160 30-second excerpts of pieces of music of highly diverse music genres but with constant tempo. Listeners were asked to tap to the beat for each excerpt. From this, a distribution of perceived tempo was generated [11]. The two highest peaks in the perceived tempo distribution for each excerpt were taken, along with their respective heights as the two tempo candidates for that particular excerpt. The height of a peak in the distribution is assumed to represent the perceptual salience of that tempo. Each algorithm participating to this MIREX task should also return two tempo candidates for each excerpt, with corresponding salience. This ground-truth data is then compared with the predicted tempo.

In 2013, our proposed model (OL) obtained the fourth<sup>3</sup> highest P-value, compared to models from 2006 to 2013, as shown in Table 1. It can be noted that these three better models are applicable only to music with stable tempo. Since then, OL has been surpassed by the two aforementioned deep-learning models [1, 2].

The current improved version of OL was submitted to the 2018 competition. The frequency resolution of the spectrogram is decreased without damaging the results. In order to filter out non-relevant peaks, the first peak at the lowest lag in the autocorrelation function is constrained to be preceded by a valley with negative autocorrelation. When comparing pairs of metrical hierarchies, only the most dominant levels of each hierarchy are selected in such

a way that we compare hierarchies with same number of levels. Finally, a periodicity that is higher than 140 BPM cannot belong to the two selected metrical levels, except if that fast pulsation is ternary, i.e., if the pulsation at the next level is three times lower. OL 2018 does not offer any improvement in the results compared to the 2013 submission.

#### 5. METRICAL DESCRIPTION

Tracking a large range of metrical levels enables to get a detailed description of the dynamic evolution of the metrical structure. For instance in Figure 3, the meter is initially and for the most part ternary. However between 40 and 50 s. (corresponding to bars 77 to 92), a little before 80 s. as well as between 120 and 130 s., we see that the ternary rhythm is actually perceived as a binary rhythm, as shown by the metrical level 0.5. Reversely in Figure 4, the meter is initially binary, but turns ternary after 80 seconds.

What is particularly interesting in those examples is also that the metrical structure changes, but the tempo remains somewhat constant. This shows that tempo is not a sufficient information for the description of metrical structure.

In order to give an indication of metrical activity that would not reduce solely on tempo but takes into consideration the activity on the various metrical levels, we introduce a new measure, called *dynamic metrical centroid*, which assesses metrical activity based on the computation of the centroid of the periods of a range of selected metrical levels, using their autocorrelation score as weight. The metrical centroid values are expressed in BPM, so that they can be compared with the tempo values also in BPM. High values for the metrical centroid indicate that more elementary metrical levels (i.e., very fast levels corresponding to very fast rhythmical values) predominate. Low values indicate on the contrary that higher metrical levels (i.e., slow pulsations corresponding to whole notes, bars, etc.) predominate. If one particular level is particularly dominant, the value of the metrical centroid naturally approaches the corresponding tempo value on that particular level.

Figure 5 shows the dynamic metrical centroid curve related to the *Allegro con fuoco* of A. Dvorak’s *New World Symphony* as shown in Figure 4. The temporal evolution of the dynamic metrical centroid clearly reflects the change of rhythmical activity between the different metrical levels, and the transition between binary and ternary rhythm, which increases the overall perceived rhythmical speed.

#### 6. DISCUSSION

The computational model OL was integrated into the version 1.6 of the open-source *Matlab* toolbox *MIRtoolbox* [12]. It also includes Goto’s aforementioned accentuation curve algorithm [7]. The updated version of OL submitted to MIREX 2018 is integrated into version 1.8 of *MIRtoolbox*.

One main limitation of all current approaches in tempo estimation and beat tracking is that the search for periodicity is carried out on a *percussive* representation of the audio recording or the score, indicating bursts of energies or spectral discontinuities due to note attacks and changes.

<sup>2</sup> <http://www.music-ir.org>

<sup>3</sup> FW already had a model in 2013 that surpassed OL.

Contestant	SB	HS	EF	FW	GK	<b>OL</b>	AK	QH	NW	DP	ES	TL	GP	FK	CD	ZG	AD	SP	MD	DE	AP	PB	GT	CB	ZL	BD
Year 20..	15	18	13	15	11	<b>13</b>	06	14	10	06	10	10	12	12	13	11	06	11	14	06	06	06	10	13	18	14
Reference	[1]	[2]					[4]										[5]									
P-score	.90	.88	.86	.83	.83	<b>.82</b>	.81	.80	.79	.78	.77	.76	.75	.75	.74	.73	.72	.71	.69	.67	.67	.63	.62	.61	.60	.54
1 tempo	.99	.98	.94	.95	.94	<b>.92</b>	.94	.92	.91	.93	.91	.89	.86	.85	.91	.82	.89	.93	.85	.79	.84	.79	.69	.85	.68	.64
both tempi	.69	.66	.69	.57	.62	<b>.57</b>	.61	.56	.50	.46	.55	.48	.61	.62	.55	.57	.46	.39	.47	.43	.48	.51	.51	.26	.46	.38

Table 1. Comparison of MIREX results from all contestants of MIREX Audio Tempo Extraction from 2006 to 2018. For each author, only the model yielding best P-score is shown. The model presented in this paper is shown in bold.

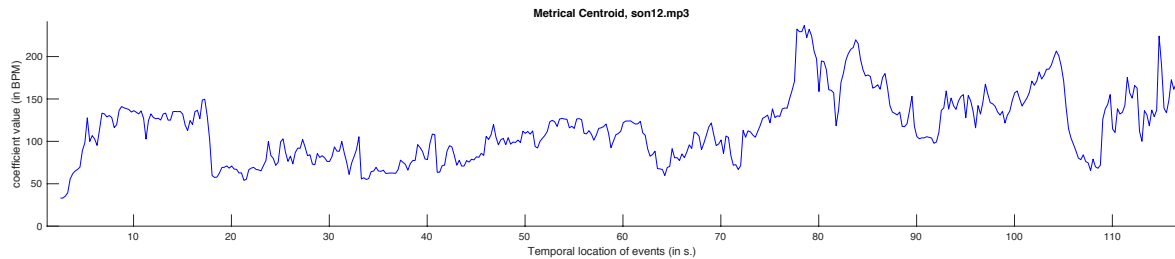


Figure 5. Dynamic metrical centroid curve for the same performance of the *Allegro con fuoco* of A. Dvorak's *New World Symphony* analysed in Figure 4.

Beyond this percussive dimension, other musical dimensions can contribute to rhythm. In particular, successive repetitions of patterns can be expressed in dimensions not necessarily conveyed percussively, such as pitch and harmony. This shows the necessity of developing methods for metrical analysis related not only to percussive regularities, but also to higher-level musicological aspects such as motivic patterns and harmonic regularities.

## Acknowledgments

This work was partially supported by the Research Council of Norway through its Centres of Excellence scheme, project number 262762. This work was also partially supported by the Swiss Center for Affective Sciences. The selection of musical materials has benefitted from collaborations with Kim Eliard and Marc-André Rappaz.

## 7. REFERENCES

- [1] S. Böck, F. Krebs, and G. Widmer, "Accurate tempo estimation based on recurrent neural networks and resonating comb filters," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2015.
- [2] H. Schreiber and M. Müller, "A single-step approach to musical tempo estimation using a convolutional neural network," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2018.
- [3] O. Lartillot, D. Cereghetti, K. Eliard, W. J. Trost, M.-A. Rappaz, and D. Grandjean, "Estimating tempo and metrical features by tracking the whole metrical hierarchy," in *3rd International Conference on Music and Emotion*, 2013.
- [4] A. P. Klapuri, A. J. Eronen, and J. T. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 11, no. 6, pp. 803–816, 2006.
- [5] M. Alonso, B. David, and G. Richard, "Tempo and beat estimation of musical signals," in *International Conference on Music Information Retrieval*, 2004.
- [6] J. P. Bello, C. Duxbury, M. Davies, and M. Sandler, "On the use of phase and energy for musical onset detection in complex domain," *IEEE Sig. Proc. Letters*, vol. 11, no. 6, 2004.
- [7] M. Goto and Y. Muraoka, "Music understanding at the beat level – real-time beat tracking for audio signals," in *IJCAI- 95 Workshop on Computational Auditory Scene Analysis*, 1996, pp. 68–75.
- [8] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *J. Acoust. Soc. Am.*, vol. 103, no. 1, pp. 558–601, 1998.
- [9] P. Toiviainen and J. S. Snyder, "Tapping to bach: Resonance-based modeling of pulse," *Music Perception*, vol. 21, no. 1, pp. 43–80, 2003.
- [10] L. V. Noorden and D. Moelants, "Resonance in the perception of musical pulse," *Journal of New Music Research*, vol. 28, no. 1, pp. 43–66, 1999.
- [11] D. Moelants and M. McKinney, "Tempo perception and musical content: What makes a piece slow, fast, or temporally ambiguous?" in *International Conference on Music Perception and Cognition*, 2004.
- [12] O. Lartillot and P. Toiviainen, "Mir in matlab (ii): A toolbox for musical feature extraction from audio," in *International Conference on Music Information Retrieval*, 2007.



# Comparison and Implementation of Data Transmission Techniques Through Analog Audio Signals in the Context of Augmented Mobile Instruments

Romain Michon,<sup>1,2</sup> Yann Orlarey,<sup>1</sup> Stéphane Letz,<sup>1</sup> and Dominique Fober<sup>1</sup>

<sup>1</sup> GRAME-CNMC, 11 Cours de Verdun-Gensoul, 69002 Lyon (France)

<sup>2</sup> Center for Computer Research in Music and Acoustics, 660 Lomita Ct., Stanford CA 94350-8180 (USA)  
rmichon@ccrma.stanford.edu

## ABSTRACT

Augmented mobile instruments combine digitally-fabricated elements, sensors, and smartphones to create novel musical instruments. Communication between the sensors and the smartphone can be challenging as there doesn't exist a universal lightweight way to connect external elements to this type of device. In this paper, we investigate the use of two techniques to transmit sensor data through the built-in audio jack input of a smartphone: digital data transmission using the Bell 202 signaling technique, and analog signal transmission using digital amplitude modulation and demodulation with Goertzel filters. We also introduce tools to implement such systems using the FAUST programming language and the Teensy development board.

## 1. INTRODUCTION

For about a decade, smartphones have been used as musical instruments [1, 2]. The fact that they combine in a single entity various sensors (e.g., accelerometers, gyroscope, touch screen, etc.), a speaker, a microphone, a battery, an Analog to Digital Converter (ADC)/Digital to Analog Converter (DAC), and a powerful processor that can be used for sound synthesis/processing make them a great platform to implement standalone Digital Musical Instruments (DMIs). However, smartphones were never designed to be used as such and they lack some crucial elements to compete with their acoustic counterparts. In previous works, we tried to solve this problem by augmenting smartphones with passive [3] and active [4] elements. While passive augmentations consist of purely acoustic elements free from electronics, active augmentations typically combine sensors, a microcontroller, and some sort of casing.

Transmitting sensor data between the microcontroller and the mobile device is often a source of problems and there currently doesn't exist a standard and comprehensive way to do this.

MIDI is commonly used for this task as it is the only

communication standard supported by all smartphones. Indeed, most external devices must be approved by Apple before they can be connected to an iPhone/iPad, but that's not the case of MIDI devices.

MIDI can be transmitted over USB or via Bluetooth. USB requires the use of a USB adapter in most cases and Bluetooth implies more complex circuitry and significantly increases the price of the augmentation.

Active augmentations also often involve the use of an external speaker/amplifier (i.e., the built-in speakers of smartphones are often weak and low quality) that needs to be connected to the audio jack of the smartphone. In that case, two cables must be plugged to the smartphone (i.e., one USB for the microcontroller and one audio cable) which is far from being an optimal solution (not to mention that more and more smartphones don't have a built-in audio jack). An alternative solution to this is to use an external USB ADC, which requires complex multiplexing operations with the microcontroller since the same USB port has to be used.

In this paper,<sup>1</sup> we present a simple lightweight solution to this problem where sensor data is transmitted to the smartphone using its audio input on its four pins audio jack (devices without a built-in audio jack can use an adapter that would be needed to retrieve the output audio signal anyway). Two techniques using respectively digital or analog data are considered:

- digital data transmission using the Bell 202 signaling technique (i.e., modem),
- analog signal transmission using digital amplitude modulation and demodulation with Goertzel filters.

The Bell 202 approach has been commonly used for transmitting digital data from an external device to a smartphone through its audio jack input. The Square Credit Card Reader<sup>2</sup> and the system presented by Kuo et al. [5] (to only cite a few) all use this technique. On the other hand, to the best of our knowledge, digital amplitude modulation has never been used in this context.

Many microcontrollers such as the ARM Cortex-M4<sup>3</sup> used on the Teensy development board series<sup>4</sup> host their

Copyright: © 2019 Romain Michon,<sup>1,2</sup> Yann Orlarey,<sup>1</sup> Stéphane Letz,<sup>1</sup> and Dominique Fober<sup>1</sup> et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> Demos and additional information about this project can be found at this URL: <https://ccrma.stanford.edu/~rmichon/analog-transmit>.

<sup>2</sup> <https://squareup.com/us/en/hardware/reader>

<sup>3</sup> <https://developer.arm.com/products/processors/cortex-m/cortex-m4>

<sup>4</sup> <https://www.pjrc.com/teensy>



Figure 1. Smartphone augmentation connection with a 4 pins audio jack.

own Pulse Width Modulation (PWM) DAC. Hence, they can be utilized to synthesize sound and play it back in real-time.<sup>5</sup> In a companion paper [6], we introduce a system where the FAUST programming language [7] can be used to program Digital Signal Processing (DSP) algorithms for the Teensy.

In the systems presented in the current paper, the Teensy DAC is used to transmit data to the smartphone. Both the Bell 202 and the Amplitude Modulation (AM) transmission techniques are implemented, evaluated, and compared.

## 2. HARDWARE

All the systems presented in this paper are based on the same hardware set-up which consists in a smartphone augmentation [4] connected to a smartphone through a 4 pins audio jack (see Figure 1). The two upper pins are used to carry the left and right audio channels out of the smartphone. The third pin (from the tip) is the ground, and the fourth pin (from the tip) carries sensor data from the microcontroller to the smartphone. The impedance between the fourth pin and the ground is 700Ω. This is important because the smartphone uses this as the trigger to activate its line input instead of using its built-in microphone. This value is also used as a reference to configure the gain of the built-in preamp.

We used a Teensy 3.2 in our system. It is based on an ARM Cortex-M4 microcontroller which hosts its own 12 bits PWM DAC running at 44.1KHz by default. The lack of reconstruction filter is compensated by the use of a 10μF capacitor connected in series between the DAC output and the fourth pin of the audio jack (see Figures 4-5). While this would not be sufficient to render a good quality audio signal, this is more than acceptable for the type of use that we make of it (see Section. 3-4).

## 3. BELL 202 SIGNALING TECHNIQUE APPROACH

The Bell 202 signaling technique allows for the serial transmission of bits at a maximum rate of 1200 baud. It uses Frequency Shift Keying (FSK) where “digital zeros” are represented by 1200 Hz tones and “digital ones” by 2200 Hz tones. Tones are typically synthesized using a square wave, so a simple digital output is theoretically sufficient to generate the corresponding analog audio signal. Instead, we preferred to use the built-in DAC of the Cortex-M4 microcontroller in order to implement a comprehensive solution only using FAUST.

<sup>5</sup> [https://www.pjrc.com/teensy/td\\_libs\\_Audio.html](https://www.pjrc.com/teensy/td_libs_Audio.html)

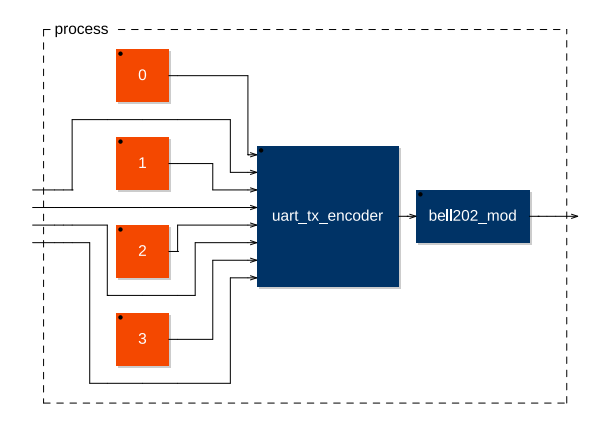


Figure 2. FAUST-generated block diagram of a program encoding four streams of data using the Bell 202 signaling technique.

### 3.1 Transmitting Data

The `bell202_mod` FAUST function<sup>6</sup> takes a stream of bits coded on audio samples and encodes it using the Bell 202 technique (essentially, the value of each bit determines/modulates the frequency of the generated square wave).

Continuous sensor values (e.g., retrieved by an analog input on the Teensy) or any other type of data can be converted to 9 bits Universal Asynchronous Receiver/Transmitter (UART) packets coded on a stream of audio samples using the `uart_tx_encoder` function (see Figure 4). `uart_tx_encoder` takes three arguments: the number of parallel streams (channels) of data to be sent, a list indicating the channel number of each stream, and a value indicating if data should be transmitted or not (one for true, zero for false). For example, the following FAUST program will send four streams of data on channels 0, 1, 2, and 3:

```
import("stdfaust.lib");
process = cm.uart_tx_encoder(4, (0,1,2,3), 1)
: cm.bell202_mod;
```

The block diagram corresponding to this FAUST program can be seen in Figure 2.

UART packets produced by `uart_tx_encoder` contain a 7 bits value, a start bit, and a stop bit. Since the channel number needs to be sent along with the corresponding value, a single value (i.e., sensor) requires a total of 18bits (two full UART packets: one for the channel number and one for the value). If the data transmission parameter of `uart_tx_encoder` (third argument) is set to false, then a stream of “one bit” is produced (that’s a standard of the UART protocol). This might be used for the potential synchronization of the sender device (i.e., Teensy) with the receiver (i.e., smartphone) in case it is plugged to it after the receiver program (i.e., app) was launched.

<sup>6</sup> All the FAUST functions presented in this paper have been added to the `communications.lib` library that can be found in the FAUST libraries repository: <https://github.com/grame-cncm/faustlibraries>.

Since bits are encoded at a rate of 1200 baud by `bell202_mod`, the bit stream produced by `uart_tx_encoder` is synchronized to that rate and several audio samples will likely contain the same bit value, depending on the audio sampling rate of the system.

`uart_tx` is a function that automatically associates FAUST User Interface (UI) elements to streams of values transmitted with `uart_tx_encoder`. Its single argument is the number of parallel stream of data (channels) to transmit. For example:

```
import("stdfaust.lib");
process = cm.uart_tx_encoder(4) : cm.
    bell202_mod;
```

sends 4 parallel streams that can be addressed on the Teensy side using the `setParamValue` method of the corresponding object generated with `faust2teensy` or `faust2api` [6]. Hence, the loop function on the Teensy could look like:

```
void loop() {
    int val0 = analogRead(A0)*127/1024;
    int val1 = analogRead(A1)*127/1024;
    int val2 = analogRead(A2)*127/1024;
    int val3 = analogRead(A3)*127/1024;
    faust.setParamValue("0",val0);
    faust.setParamValue("1",val1);
    faust.setParamValue("2",val2);
    faust.setParamValue("3",val3);
}
```

where `faust` is a FAUST object produced with `faust2api` [6], and the first argument of the `setParamValue` method the FAUST parameter name automatically generated by the `uart_tx` function corresponding to the channel number on which the value should be transmitted.

Note that it is also possible to write a FAUST program to carry out the same task without writing a single line of Arduino code using `faust2teensy` [6]. In that case, analog and digital inputs of the Teensy can be mapped to FAUST UI elements using metadata:

```
val0 = nentry("val0[io: A0]",0,0,127,1);
process = val0 :
    cm.uart_tx_encoder(1,(0),1) :
    cm.bell202_mod;
```

### 3.2 Receiving Data

Data transmitted by the microcontroller using the technique presented in Section. 3.1 can be decoded directly in the FAUST program implementing the app running on the smartphone (see Figure 4). This app was generated with `faust2smartkeyb` [8]. The `bell202_demod` function can be used to decode the signal produced by `bell202_mod` on the Teensy to turn it into a stream of bits encoded on a digital audio signal. The decoding algorithm uses zero crossing detection and cross-correlation. `bell202_mod` and `bell202_demod` are configured to have the same baud so they don't need to be parametrized.

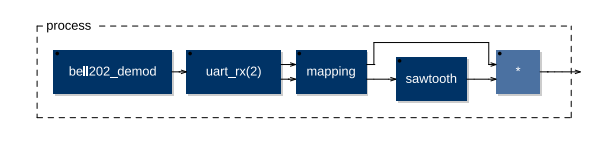


Figure 3. FAUST-generated block diagram of a program decoding two streams of data using the Bell 202 signaling technique.

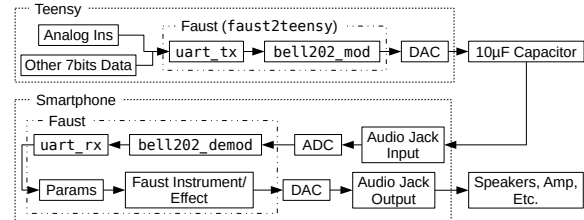


Figure 4. Analog audio sensor data transmission between a Teensy and a smartphone using the Bell 202 Signaling Technique.

The `uart_rx` function takes a single argument allowing us to specify the number of channels to be extracted from the input bit stream. This number should be the same as the one used with `uart_tx_encode`. `uart_rx` outputs audio signals (one per channel) containing the transmitted data for each individual channel. These signals (whose range is 0-127) can be used directly to control some sound synthesis/processing parameter. In the following example, two sensor data streams are decoded and used to control the gain and the frequency of a sawtooth wave oscillator:

```
import("stdfaust.lib");
mapping = /(127),/(127)*1900 + 100);
process =
    cm.bell202_demod : cm.uart_rx(2) :
    mapping : *(os.sawtooth);
```

## 4. DIGITAL AMPLITUDE MODULATION APPROACH

This other method consists in carrying continuous sensor signals on AM bands to the smartphone.

### 4.1 Transmitting Data

`am_tx_encoder` is a FAUST function working in a similar way than the combination of `uart_tx_encoder` and `bell202_mod` (see Section. 3.1). Its first argument configures the number of parallel streams to be transmitted and its second argument is a list of channel numbers corresponding to each data stream input. For example, the following FAUST program will send four streams of data on channels 0, 1, 2, and 3:

```
import("stdfaust.lib");
process = cm.am_tx_encoder(4,(0,1,2,3));
```

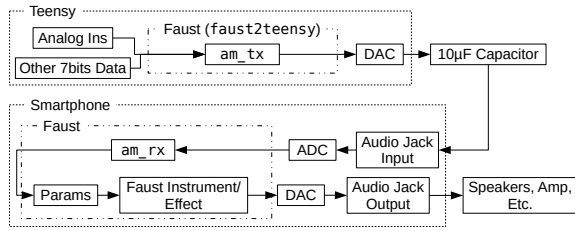


Figure 5. Analog audio sensor data transmission between a Teensy and a smartphone using Amplitude Modulation.

One sine wave oscillator is used for each channel. Their gain is modulated by individual sensor data. The frequency of each oscillator is determined by the number of bands. They are distributed between 50 and 20000 Hz (once again, the sampling rate of the DAC on the Teensy is 44.1 KHz). Of course, the gain of each sine oscillator is scaled in function of the number of bands, which means that more bands means less dynamic range. In addition to that, a calibration sine tone with maximum gain is constantly transmitted at 20 KHz to normalize the input signal on the smartphone side in real-time. Hence, if four sensor streams are transmitted, five sine tones will be generated.

`am_tx` works the same way than `uart_tx` and takes a single argument which is the number of channels to be transmitted. User interface elements are automatically generated by this function and can be addressed on the Teensy side using the `setParamValue` method (see Section. 3.1).

## 4.2 Receiving Data

Data encoded by the `am_tx` function on the Teensy can be decoded using `am_rx` on the smartphone. This function takes a single argument which corresponds to the number of data streams to be decoded/demodulated. This number should be the same as the one used with `am_tx` on the Teensy. `am_rx` outputs audio signals (one per channel) containing the transmitted data for each individual channel. These signals (whose range is 0-127) can be used directly to control some sound synthesis/processing parameter. In the following example, two sensor data streams are decoded and used to control the gain and the frequency of a sawtooth wave oscillator:

```
import("stdfaust.lib");
mapping = /(127), /(127)*1900 + 100);
process =
  cm.am_rx(2) : mapping : *(os.sawtooth);
```

`am_rx` uses Goertzel filters [9] to efficiently extract the amplitude of each band. The filters block/window size is automatically adapted in function of the number of bands to be decoded. A greater block size will provide more precision in the frequency domain but will add more latency. Hence, more bands means more delay (see Section. 5).

Other demodulation techniques were tested (e.g., band-pass filters, etc.) but Goertzel filters provided the best results in this context.

## 5. EVALUATION

Table 1 compares the performance of the data transmission techniques presented in Section. 3 and Section. 4 for various numbers of parallel channels. Latency and bit depth are the two main parameters that are considered.

A single channel transmitted with the 202 technique corresponds to a latency of  $\sim 15\text{ms}$  (18 bits are required to transmit one value at a bit rate of 1200 bits/s so  $\frac{1}{(1200/18)}$ ). Hence, every new channel will add a latency of  $\sim 15\text{ms}$ . On the other hand, latency when using the AM technique is determined by the block size of the Goertzel filter which is automatically computed by `am_tx_encoder` in function of the number of channels (see Section. 4.2). Note that overall, this method provides much better latency performances than the 202 approach.

The bit depth of the data transmitted with the 202 technique is constant (7 bits by default) and can be decided by the programmer. A greater bit depth means more precision but will also add more latency. For the AM technique, the precision/range of the data depends on the number of channels to be transmitted. Since the built-in DAC of the Cortex-M4 can produce 12 bits values, the range of data when transmitting a single channel is 2048 ( $2^{12}/2$  where the division by two corresponds to the number of carriers: here one for the data and one for the calibration signal). In practice, the range of the data for the AM technique is probably slightly smaller because of potential noise in the signal, but this type of parameter is hard to measure efficiently.

The Goertzel filter demodulation approach used with the AM transmission technique has proven very effective. Also, since the carriers are modulated at a rate of approximately 440 Hz, sidebands are not an issue as long as the frequency of each carrier is more than 880 Hz apart (i.e., if more than 20 channels were needed this rate could be lowered, etc.).

All in all, both techniques present advantages. The 202 approach is obviously more reliable but it is also less powerful than the AM method, especially when a large number of parallel streams of data must be transmitted. It is also technically more complex to implement.

## 6. CONCLUSIONS

The built-in analog audio input of smartphones provides a convenient and standard way to acquire sensor data from a microcontroller to control sound synthesis/processing parameters. This is very helpful in the context of active smartphone augmentations where “prosthetics” are mounted on the device to expand its affordances.

In this paper, two transmission techniques usable in this context as well as their associated tools were presented and compared. They seamlessly integrate to the existing panoply of FAUST-based tools to create musical instruments with augmented smartphones.

We believe that this approach solves various issues by providing a standard universal way to connect to smartphones and by offering better performances than other standards such as USB/Bluetooth MIDI, etc.



N Channels	202 Latency	AM Latency	AM Goertzel Block Size	202 Range	AM Range
1	~15ms	~6ms	256	128 (7 bits)	2048
2	~30ms	~6ms	256	128 (7 bits)	1365
3	~45ms	~11ms	512	128 (7 bits)	1024
4	~60ms	~11ms	512	128 (7 bits)	818
5	~75ms	~23ms	1024	128 (7 bits)	683
10	~150ms	~23ms	1024	128 (7 bits)	372
15	~225ms	~46ms	2048	128 (7 bits)	256
20	~300ms	~46ms	2048	128 (7 bits)	204

Table 1. Comparison of the Bell 202 signaling technique with the amplitude modulation transmission approach.

## 7. REFERENCES

*Digital Signal Processing*, vol. 48, no. 7, pp. 691–700, 2001.

- [1] G. Essl and M. Rohs, “Interactivity for mobile music-making,” *Organised Sound*, vol. 14, no. 2, pp. 197–207, 2009.
- [2] G. Wang, “Ocarina: Designing the iPhone’s Magic Flute,” *Computer Music Journal*, vol. 38, no. 2, pp. 8–21, Summer 2014.
- [3] R. Michon, J. Smith, M. Wright, C. Chafe, J. Granzow, and G. Wang, “Passively augmenting mobile devices towards hybrid musical instrument design,” in *Proceedings on the New Interfaces for Musical Expression Conference (NIME-17)*, Copenhagen, Denmark, May 2017.
- [4] R. Michon, J. O. Smith, M. Wright, C. Chafe, J. Granzow, and G. Wang, “Mobile music, sensors, physical modeling, and digital fabrication: Articulating the augmented mobile instrument,” *Applied Sciences*, vol. 7, no. 12, p. 1311, 2017.
- [5] Y.-S. Kuo, T. Schmid, and P. Dutta, “Hijacking power and bandwidth from the mobile phone’s audio interface,” in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, Austin, Texas, 2010.
- [6] R. Michon, Y. Orlarey, S. Letz, and D. Fober, “Real time audio digital signal processing with Faust and the Teensy,” in *Proceedings of the Sound and Music Computing Conference (SMC-19)*, Malaga, Spain, 2019.
- [7] Y. Orlarey, S. Letz, and D. Fober, *New Computational Paradigms for Computer Music*. Paris, France: Delatour, 2009, ch. “Faust: an Efficient Functional Approach to DSP Programming”.
- [8] R. Michon, J. Smith, C. Chafe, G. Wang, and M. Wright, “faust2smartkeyb: a tool to make mobile instruments focusing on skills transfer in the Faust programming language,” in *Proceedings of the International Faust Conference (IFC-18)*, Mainz, Germany, July 2018.
- [9] R. Beck, A. G. Dempster, and I. Kale, “Finite-precision Goertzel filters used for signal tone detection,” *IEEE Transactions on Circuits and Systems II: Analog and*

# MASS-INTERACTION PHYSICAL MODELS FOR SOUND AND MULTI-SENSORY CREATION : STARTING ANEW

**Jerome Villeneuve**

Univ. Grenoble Alpes, CNRS, Grenoble INP\*  
GIPSA-Lab, 38000 Grenoble, France  
jerome.villeneuve@gipsa-lab.fr

**James Leonard**

Univ. Grenoble Alpes, CNRS, Grenoble INP\*  
GIPSA-Lab, 38000 Grenoble, France  
james.leonard@gipsa-lab.fr

## ABSTRACT

Mass-interaction methods for sound synthesis, and more generally for digital artistic creation, have been studied and explored for over three decades, by a multitude of researchers and artists. However, for a number of reasons this research has remained rather confidential, subsequently overlooked and often considered as the *odd-one-out* of physically-based synthesis methods, of which many have grown exponentially in popularity over the last ten years. In the context of a renewed research effort led by the authors on this topic, this paper aims to reposition mass-interaction physical modelling in the contemporary fields of Sound and Music Computing and Digital Arts: what are the core concepts? The end goals? And more importantly, which relevant perspectives can be foreseen in this current day and age? Backed by recent developments and experimental results, including 3D mass-interaction modelling and emerging non-linear effects, this proposed reflection casts a first canvas for an active, and resolutely outreaching, research on mass-interaction physical modelling for the arts.

## 1. INTRODUCTION

This paper intends to express a refreshed vision on the use of mass-interaction modelling for real-time sound-synthesis and interactive digital arts. After positioning some general concepts, we briefly document how a step aside from sound-based considerations has led to new grounds for investigating the potential of mass-interaction physical modelling. We then present multi-dimensional geometry as a starting point for any kind of mass-interaction model (in terms of mathematical roots, modelling methodologies and performances), and finally discuss the relevance of this approach for modelling and real-time simulation of virtual acoustical structures that present emergent non-linear behaviour.

## 2. PHYSICAL MODELLING: WHY BOTHER ?

When considering the term physical modelling as it is used in a large number of fields of research, the first and most fundamental question to arise is :

\*Institute of Engineering, Univ. Grenoble Alpes.

Copyright: © 2019 Jerome Villeneuve et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

*Why would one painstakingly reproduce the behaviour (phenomenological approach) or a virtual representation (causal approach) of a real physical object ?*

In most of these fields the answer will have to do with saving lives, or a significant increase in the understanding of our world, from quarks to universe(s?). But when it comes to sound synthesis, this question might be slightly more difficult to answer as it becomes:

*Why reproduce - and try to play - a virtual instrument that mimics a real instrument that one could play in real life?*

Over the years, the Computer Music community has developed very serious and meaningful arguments justifying this approach, such as those recalled by Bilbao and Smith in the opening chapters of their respective books [1, 2]. Resulting research has subsequently led to the development of a rich variety of techniques [3–5], and continues doing so to this day. However, if we allow ourselves to take a more poetic stance, the above question could subjectively be qualified as irrelevant, to the benefit of the following :

*How could our most common day-to-day physical experience of the physical world inspire and ease an artistic process in its digital counterpart ?*

The entire approach described hereafter takes ground here. Although it is anchored in a physical paradigm, its main focus is not the common “realistic sound certified by impeccable evaluation methodology” achievement. It approaches physically-based synthesis from a different angle, oriented mostly towards intuitive, interactive and multisensory exploration - without excluding methodological questioning of its validity along the way. Here, physical modelling consists in designing and experimenting virtual mechanical constructions, with the aim of discovering and crafting a range of uncanny sound-producing objects that can be directly explored and interacted with.

## 3. A WORLD OF INTERACTING MASSES

All approaches for modelling the physics of macro-scale mechanical systems stem from a common root: Newton and his laws. The mass-interaction (MI) paradigm is one out of many ways to transcribe these laws into discrete time and space algorithms that allow for the computation of physical dynamics. It does so by representing physical models as networks of mass-type elements and interaction-type elements [6].

### 3.1 A balancing act

While MI is probably not the most computationally effective physical modelling approach, nor the most elegant in terms of mathematical formulation, and - let's face it - not the most suited paradigm for synthesising supra-realistic sounds of legendary acoustic instruments, it possesses four undeniable qualities: genericity, modularity, ease of use given an intuitive understanding of physics, and very direct possibilities for gestural interaction [7]. These traits are especially important when considering the perspective of making sound synthesis tools accessible to everyone.

Ultimately, our choice to further investigate mass interaction physical modelling is motivated by the conviction that it strikes a “good balance”: a generally satisfying compromise that still yields strong potential regarding, on the one hand, scientific and technological considerations, and on the other hand, artistic and creative perspectives. And while such a balance is not always simple to maintain, it is central to the authors’ research methodology.

### 3.2 Modular Physical Modelling

#### 3.2.1 Modularity in Artistic Creation

Under general consideration, modularity can qualify any physical or abstract system regarding the capacity of its irreducible elements to connect to each other and thus achieve more complex objects or functionalities (cf. fig.1). Max Mathews [8] said of modularity concept (referring to unit generator in MUSIC III) :

*“It’s a very important concept, and more subtle than it appears on the surface. I wanted to give the musician a great deal of power and generality [...], but at the same time I wanted as simple a program as possible; I wanted the complexity of the program to vary with the complexity of the musician’s desires. [...] The only answer I could see was not to make the instruments myself [...] but rather to make a set of fairly universal building blocks and give the musician both the task and the freedom to put these together into his or her instruments.”*

This concept is specifically valued in recreational contexts (cf. fig.2) and creative fields. As examples of modular systems in musical creation, one could cite modular analog sound synthesis or patching environments such as Max/MSP, PureData and Chuck. Regarding other creative fields, such as graphic design and video, one could think of Processing<sup>1</sup>, Quartz Composer, vvvv<sup>2</sup>, and more recently NodeBox<sup>3</sup>. All of which have led to the emergence of novel artistic processes and to the development of important user communities.

#### 3.2.2 A modular approach to physical modelling

In the scope of this paper, modularity is the most fundamental *a priori* that the authors will maintain. It is in fact the central pivot around which revolve all the efforts to find true meaning to mass-interaction physical modelling. It is the necessary condition to achieve a vast range of emerging

<sup>1</sup> <https://processing.org>

<sup>2</sup> <https://vvvv.org>

<sup>3</sup> <https://www.nodebox.net>

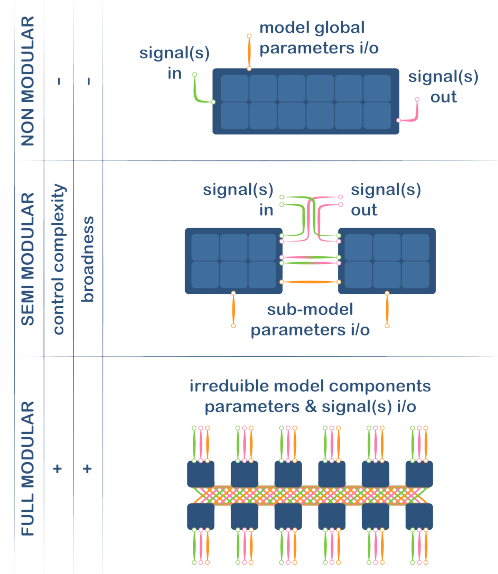


Figure 1. Different degrees of modularity. Modular approaches entail higher control complexity but offer a broader spectrum of possible outcomes and results.

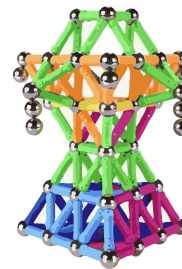


Figure 2. A modular mechanical construction paradigm, based on two types of elements ( “node” and “links”).

behaviours, and to be struck by surprise (and we are not referring to numerical instability !) each time the smallest part of a model is modified.

Of course, such promising perspectives cannot be expected without a challenging counterpart. It might be a little bit more complex (what a reassuring euphemism) to handle fully modular rather than non-modular approaches. But, if one gives it a little thought, even if the learning curve can be steep, fully-modular systems can always be approached by starting from the simplest possible combination of elements and set of parameters. From there, one can - in his/her own good time - progressively build a knowledge base guided mostly by elementary concepts, in our case relating to mechanical physics.

#### 3.2.3 Does modular physical modelling really pay off?

One might think that this enthusiasm for modularity could be curbed by the involved computational complexity, as it generally limits possibilities for optimisation or algorithmic “shortcuts”. However, with regard to the state of the art of physical modelling approaches (in terms of computation, richness of sounds, dealing with non-linearities, etc.) and

given recent sound synthesis experimentation with 3D mass-interaction models described hereafter, the position of mass-interaction physical modelling appears relatively solid.

Moreover, when adopting the modular “creative” modelling mindset, complexity and computation can be leveraged by a number of factors. Indeed, while model scale does have a notable impact (especially for structures such as dense plates, e.g. the cymbal), there is a good chance that the sonic essence you (maybe didn’t even know that you) were looking for can *in fact* be obtained with a simpler model than the physical system you *would have* imagined<sup>4</sup>, resulting in much simpler computation than the discretised solution to a complex mathematical representation of such a system.

Finally, all of the above considerations bring about yet another question:

*If a modular physical modelling paradigm can faithfully<sup>5</sup> represent any physical system that can be described by tridimensional point-based Newtonian mechanics, shouldn’t it naturally bring forth many of the emergent non-linear characteristics found precisely in the spatial and physical structure of such systems?*

If so, could such an approach to non-linearities, crucial factors in the richness of synthesised sounds [9], be easier to apprehend, manipulate and pass on to artists than the common route of advanced mathematical formulations and finite difference schemes? This discussion will be for a later section.

### 3.3 Here is the motto

In short, the authors still see mass-interaction physical modelling as a very fertile ground to explore, especially under the prism of digital artistic creation. However, unlike much previous work and already significant discoveries made on the subject [6, 10, 11], the motto here has to be explicitly clarified :

1. Each and every core concept, component or experimental paradigm must be openly stated, positioned and questioned within a global scientific framework.
2. Every model, each line of code, will be shared so as to allow for a community-driven artistic, scientific and technological reflection regarding this topic.

## 4. OBSERVE, THEN INTERACT, THEN - AND ONLY THEN - LISTEN

### 4.1 Stepping aside from a sound-based approach

The mass-interaction (MI) approach presented here allows to address the modelling question from any kind of preliminary phenomenological consideration. In simpler terms, one could create a virtual physical model with the sole aim to observe a visual rendering of its motion through time, or with the intent to explore the properties of an interaction

<sup>4</sup> see section 6 for an example regarding chaotic emergence generally associated with cymbal-like structures - obtained with a relatively small model that is nothing like a cymbal.

<sup>5</sup> well, within the limits of numerical stability.

model, or further still considering the motion of a virtual mechanical deformation as a sound source. Ultimately, every single object designed in MI with a specific idea and modality in mind can be considered (as it is) for its complementary modalities.

This property gives MI a strong potential in the fields of sound synthesis, interaction modelling, visual rendering, and to create multisensory virtual objects. Of course, the latter raises questions as to the various contexts in which one can build and simulate such objects.

The following section describes a scenario followed by the authors, stemming from visual considerations in a visual rendering software, and progressively leading to explore the resulting objects for their acoustical properties and playability (including via Haptic interaction) - all within this visual rendering software.

### 4.2 Computing and rendering mechanical motion

Mass-interaction physical modelling has long been studied and used within the domain of visual arts<sup>6</sup>, resulting in a string of concepts and tools [12, 13]. On the basis of these visual considerations and the will to increase accessibility to MI modelling through open-source software, the authors recently developed *miPhysics*, a compact JAVA library then targeted essentially for the Processing environment (a software sketchbook & language widely used for prototyping and creating visual and interactive arts). This tool was naturally written to allow designing and computing the motion of point-based models, described as an arrangement of masses and interactions in up to 3 spatial dimensions, and each mass possessing up to 3 degrees of freedom.

The flexibility of a framework such as Processing allows for efficient interactive modelling and naturally leads to consider aspects such as real-time parameter control, *on-the-fly* topology/geometry alterations, as well as numerous rendering methods for large models. Despite the fact that it is, by essence, a prototyping environment (therefore not necessarily well optimised for complex scene rendering), large-scale models composed of tens of thousands - and sometimes over a hundred thousand - elements run in real-time (cf. fig.3) at physics computation rates from 250 Hz up to several kHz, and visual display rates around 60 FPS.

### 4.3 From motion and physical interaction to sound

While observing the visual motion such rendered models, a recurring interrogation quickly became: “wow, how would that *sound*?”. Unable to contain ourselves, we then started cranking simulation rates up to 44.1kHz in an audio thread, connecting “microphones” into these virtual scenes in the simplest way possible (applying the motion of one or more mass elements directly to a loudspeaker - direct output from *very* localised listening points with no considerations of sound propagation through an aerial medium), and there you have it: multisensory sound and visual objects at your fingertips, directly within Processing (cf. fig.4).

...well, as such the last statement is not strictly true - actually *touching* these objects requires force-feedback inter-

<sup>6</sup> see for instance A. Mondot and Claire B.’s creation, Hakanai.



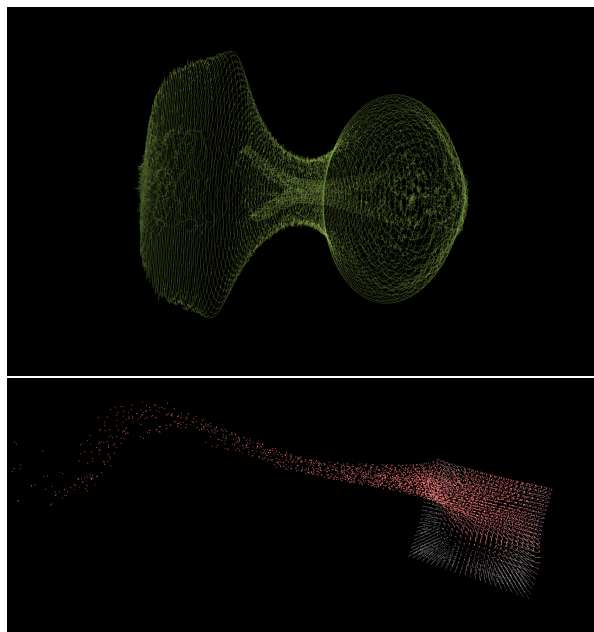


Figure 3. Snapshots of real-time *miPhysics* models running in Processing at 250Hz. The first model counts 100000 modules (50k masses, 50k interactions). The second counts 60000 modules (15k masses, 45k interactions).

action. This was integrated using the Haply<sup>7</sup> open-source device (shown in fig.5) as detailed in [14].

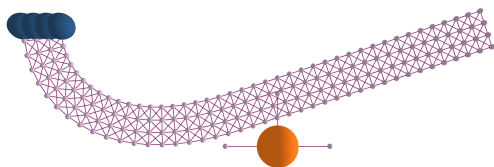


Figure 4. Snapshot of a *miPhysics* real-time model running in Processing at 44.1kHz, containing 708 modules (172 masses, 536 interactions) and playable by mouse control.

This leaves us with an entirely open environment in which virtual objects can be synthesised in real time in a 3 dimensions and 3 degrees of freedom space, and are accessible to all of our senses, save for smell and taste. This constitutes a result in itself, especially given that many of today's aspects of sound and music research (interaction mapping, model visual rendering, VR-reinforced presence...) address larger considerations than sound alone.

But that's not all<sup>8</sup>, since another historical [15] and still very actual state of the art problem in sound synthesis [9] naturally finds some solutions when Newton's equations are finally given some space<sup>9</sup>: non-linearities.

<sup>7</sup> <http://www.haply.co/>

<sup>8</sup> insert synthesised sound of drum rolls

<sup>9</sup> insert synthesised sound of a heavily-struck cymbal, from which a vast panoply of non-linear behaviours emerge

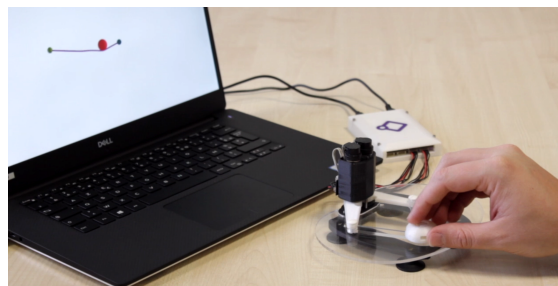


Figure 5. Direct haptic interaction with a mass-interaction model of a 3D string using the Haply device.

## 5. 3D MASS-INTERACTION 101

### 5.1 The grizzly details

Now that we have presented our position in regards to a research dedicated to mass-interaction physics and teased at some early results, it seems a good time to introduce (or reintroduce) the scientific and technological concepts behind mass-interaction physical modelling.

#### 5.1.1 Have you met Newton?

A little while back, a fine gentleman by the name of Isaac Newton stated the following:

1. *In an inertial frame of reference, an object either remains at rest or continues to move at a constant velocity, unless acted upon by a force.*
2. *In an inertial frame of reference, the vector sum of the forces  $f$  on an object is equal to the mass  $m$  of that object multiplied by the acceleration  $a$  of the object:  $f = ma$ .*
3. *When one body exerts a force on a second body, the second body simultaneously exerts a force equal in magnitude and opposite in direction on the first body.*

These three rules are the basis for resolving just about any mechanical system, by representing it as punctual masses and by expressing different kinds of forces (gravitational, elastic, frictional, etc.) applied to them.

#### 5.1.2 A numerical discretisation scheme

As in any numerical resolution to a set of partial difference equations, various discretisation schemes may be employed, from lower order methods (such as Euler) to more complex schemes (such as Runge-Kutta). The choice of a scheme results from considerations of numerical stability, computational complexity and causality.

If finite-difference schemes for lumped methods in the 1D case are well documented in physical modelling literature (see [1, 5, 6]), the  $N$ -dimensional case has rarely been a topic of interest within this community. Below, we present a formulation in which positions and forces are  $N$ -D vectors (3D in the case of the *miPhysics* library).

A common starting point for representing and computing discretised modular mass-interaction systems<sup>10</sup> is to apply a second order central difference scheme to Newton's second law:

$$\mathbf{f} = m \cdot \mathbf{a} = m \cdot d^2 \mathbf{u} / dt^2 \quad (1)$$

where  $\mathbf{f}$  is the force applied to the mass,  $m$  is its inertia,  $\mathbf{a}$  its acceleration vector and  $\mathbf{u}$  its position vector. It results in the following normalised form where discrete-time position and force vectors are noted  $\mathbf{U}$  and  $\mathbf{F}$ ,  $M$  is the discrete time inertial parameter defined as  $M = m / \Delta T^2$ , and  $\Delta T$  is the sampling interval:

$$\mathbf{U}_{(n+1)} = 2 \cdot \mathbf{U}_{(n)} - \mathbf{U}_{(n-1)} + \frac{\mathbf{F}_{(n)}}{M} \quad (2)$$

This leaves interactions (the elements that apply forces to material points). In most cases of mechanical interactions, the force exerted can be expressed as a function of position and velocity: as an example, the magnitude of a visco-elastic force applied by a linear spring (with stiffness coefficient  $k$ , damping coefficient  $z$  and resting length of  $l_0$ ) connecting a mass  $m_2$  at the position  $u_2$  to a mass  $m_1$  at the position  $u_1$  is given by:

$$f_{1 \rightarrow 2} = -k \cdot (||\mathbf{u}_2 - \mathbf{u}_1|| - l_0) - z \cdot (||\mathbf{v}_2 - \mathbf{v}_1||) \quad (3)$$

Approximating the velocity with the backward Euler scheme, we obtain  $F_{spring}$  the force scalar value:

$$\begin{aligned} dist_{(n)} &= ||\mathbf{U}_{2(n)} - \mathbf{U}_{1(n)}|| \\ F_{spring(n)} &= -K(dist_{(n)} - l_0) \\ &\quad - Z \cdot (dist_{(n)} - dist_{(n-1)}) \end{aligned} \quad (4)$$

With the discrete-time stiffness parameter  $K = k$ , and the discrete-time damping parameter  $Z = z / \Delta T$ . The resulting force vector (defined along the direction vector between both masses) is finally applied symmetrically onto each mass (Newton's third law):

$$\begin{aligned} \mathbf{F}_{proj(n)} &= F_{spring(n)} \cdot \frac{\mathbf{U}_{2(n)} - \mathbf{U}_{1(n)}}{||\mathbf{U}_{2(n)} - \mathbf{U}_{1(n)}||} \\ \mathbf{F}_{2 \rightarrow 1(n)} &= -\mathbf{F}_{proj(n)} \\ \mathbf{F}_{1 \rightarrow 2(n)} &= +\mathbf{F}_{proj(n)} \end{aligned} \quad (5)$$

The main difference in regards to the classical "topological" 1D algorithms is the explicit use of Euclidian geometry associated to the spatial attributes of the physical mass-type elements.

### 5.1.3 Computing the system dynamics

A step of the discrete physical computation is structured as follows: masses compute their new positions according to the discrete-time vector sum of forces  $F$ , resulting in an update of the model positions. Interactions then calculate applied forces (using the newly calculated positions and positions from the previous step), resulting in new sum vector forces for each mass. This is used in the next computation step for the calculation of new mass positions, and so forth.

<sup>10</sup> as in the CORDIS ANIMA formalism.

### 5.1.4 Modules and properties

Mass modules are defined by an inertia parameter (possibly infinite, hence representing anchored points) and a set of spatial coordinates and speed initial values.

Interaction modules are defined by one or both stiffness and viscosity parameters and a resting length. These modules can also include conditional proprieties naturally leading to non-linear interactions (such as representing contact forces between material elements).

## 5.2 Modelling

Based on the previous elementary concepts and modules, modelling with MI consists in building a geometrical model by positioning and connecting material components together through interactions components, and by specifying the parameters and initial conditions of each one. Furthermore, every consideration regarding ways of listening, visualising or interacting with one or several modules, parameters, or even with topological or geometrical properties of a model, are up to the user.

### 5.2.1 Do not fear simplicity

Given that models can contain tens of thousands of elements, the perspective of configuring physical parameters and initial states can seem a little daunting. When exploring larger structures a simple first approach is to consider locally homogeneous parameters in sections of the object, as they can always be fine-tuned at a later stage. However, it is worth noting that richer mass-interaction behaviour does not always stem from huge homogeneous models (which take on a role similar to a propagation medium) but often somewhere in the middleground - where careful parameter tuning and interaction design meets with sufficiently rich resonant structure to catch our ear ! More generally, users can deploy several strategies to build up their models, either from scratch one module at a time, or through scripting strategies for geometry or multi-parameter specification.

### 5.2.2 Figuring out 3D

Even though clear parallels can be drawn between mass-interaction methods and those of general finite differences and/or finite element methods, the most notable difference lays in the way matter and spaciality are considered. FDM/FEM methods discretise unidimensional or multidimensional objects, reducing them to numerous small sections of linear, planar or volumetric geometries. Each section is literally anchored to a static position in space around which the matter it represents will evolve locally, allowing wave propagation. The latter can occur along one or several dimensions, while, in most cases, each local section of space allows the matter it represents to move along one degree of freedom. But at the global scale, the general geometry of these objects will never ever change.

On the other hand, 3D mass-interaction do not pre-suppose any spatial grid or subsections that virtual matter can be hung to. In MI, masses are free to go wherever they see fit (and they can move along 3 degrees of freedom!). Hence, the interactions connecting them, and their properties (particularly in terms of resting length), are crucial. Ultimately,

each and every module contributes to the global materiality of an object, considering both its geometry (and structural consistency) and its mechanical properties.

In simpler terms, this means that if one creates a cube with 8 masses and 12 interactions, the slightest blow on it will make it collapse in on itself : such a cubic mesh is insufficient to describe a structurally consistent cubic virtual object (cf. fig.6). You will have to consolidate it and more generally think “deformable solid” (think of it as if fig.2’s links could be elastic).

### 5.3 Efficiency

Even if the 3D mass-interaction engine implemented in Processing must be regarded as a non-optimised prototype, it allows to seize the potential of such a method. Figure 7 gathers four scenarios of real-time models, referencing model complexity (number of mass-type and interaction-type elements) and involved modalities (visual, audio and/or haptic). Performance was measured on a single core of a standard laptop<sup>11</sup> : models pass if there are no image, audio or haptic dropouts<sup>12</sup> during computation. The general stability of *miPhysics* is not an obstacle to the wildest experiments. Plus, its boundaries are very well known and understandable by any user.

## 6. NON-LINEAR BEHAVIOURS !

### 6.1 Do the math - or maybe don’t

Within the last 20 years, the emphasis of physically-based sound synthesis has shifted from exciting (generally) linear resonators via non-linear interactions towards replicating complex acoustical behaviours through the modelling of non-linear dynamics in resonating structures. Modal synthesis [16], Waveguide methods [17], FDTD and even 1D mass-interaction systems [18] have seen themselves reinvented to this end.

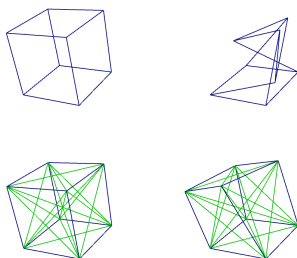


Figure 6. Top: a rudimentary MI cubic mesh, and bottom: a more complex mesh considering “structural consistency”. Left: initial state, right: state after a little push. The bottom cube jiggles around and progressively returns to its initial configuration, whereas the top cube collapses instantly.

Conversely, recent years show a strong increase in finite-difference-time-domain schemes, as computational limitations lessen, now allowing for off-line synthesis of very

<sup>11</sup> Dell Precision 5530 running Ubuntu 18.04 & Processing 3.5.3, Specs: Intel i7-8850H 4 cores at 2.6GHz, 16GB RAM.

<sup>12</sup> since the OS is non real-time, sporadic missed haptic frames can occur [14] - although not enough to significantly alter haptic interaction.

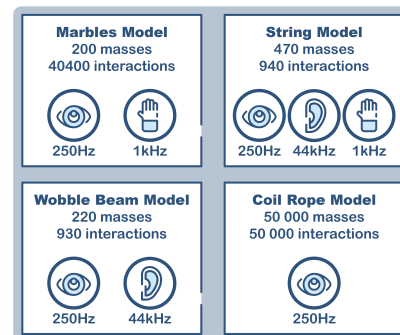


Figure 7. A quick overview of 3D mass-interaction real-time capabilities regarding several combinations of auditory, visual and haptic modalities.

large models, as well as real-time synthesis of small to medium scale ones. In the dominant literature, systems are first formalised under linear conditions (from the 1D, 2D, or more rarely 3D, wave equation - and often considering vibrations along a single dimension) before adding specific non-linear formulations to account for phenomena such as “airy stress”, leading to effects such as pitch glide and chaotic oscillations in plates [1].

In all of the above, non-linearities present themselves as mathematical ramifications, incorporated into formulations primarily rooted in acoustics - taking modal representations outside of their comfort zone, one could say.

We propose the *complete* opposite: within a framework that fully accounts for the tri-dimensional spatial properties of matter (cf. above), build vibrating bodies, give them a good smack, and observe. If Newton was right, the pandora box of non-linear behaviour might just open - without us ever having to write an equation for it<sup>13</sup>.

### 6.2 Experiments & Observations

#### 6.2.1 What can we expect?

Many sonic attributes can be attributed to non-linear phenomena in vibrating bodies. Fletcher and Rossing [19] mention : Dependence of vibration mode frequencies upon amplitude of excitation (equivalent to “tension modulation”), the generation of overtones that are exact harmonics of the fundamental oscillation (“harmonic distortion”), forced oscillations at submultiples of the driving frequency, and chaotic oscillations. Below, we present results from simple 3D mass-interaction models, with the aim of (hopefully!) observing some of these phenomena.

#### 6.2.2 Observation of tension modulation

Experimental measures (cf. fig.9) were conducted on a simulated string, composed of 32 masses, excited at 1/3 of its length by varying levels of force impulse. The resulting spectrogram: The pitch glide generated by the tension modulation is immediately apparent, and correlated to the excitation amplitude.

<sup>13</sup> us mass-interaction people hate writing equations.



Indeed, the purely linear springs exert a recall force proportional to the Euclidian distance between masses and are therefore dynamically affected by elongation (compression or distension of the spring): larger excitation means more elongated springs, resulting in increased tension.



Figure 8. 3D *miPhysics* string model counting 32 masses and 31 interactions.

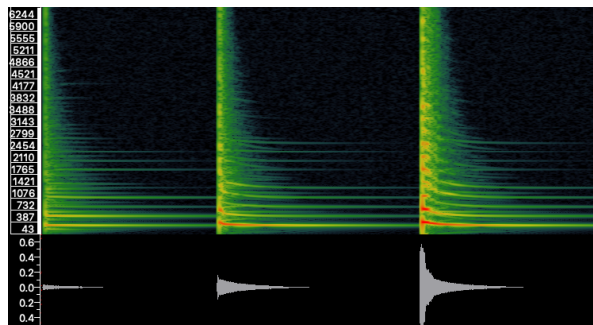


Figure 9. Emerging effect of tension modulation related to an increasing excitation amplitude, with string model of fig.8 running in real-time at 44.1kHz.

### 6.2.3 Observation of chaotic oscillations

In this case, the physical model is a 3D beam (86 masses and 512 interactions, based on the “structurally consistent” cubes of fig.6) fixed at both ends and struck by a “plucking” mechanism with varying levels of speed (cf. fig.10).

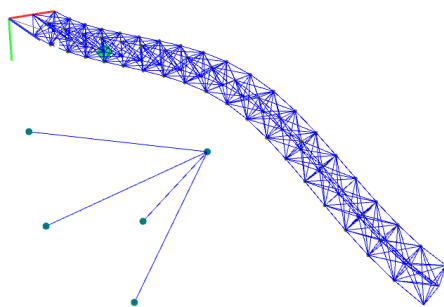


Figure 10. 3D *miPhysics* model of a beam counting 86 masses and 512 interactions, running in real-time at 44.1kHz, excited by a plucking mechanism.

Figure 11 shows, on the one hand a low amplitude excitation, resulting in clear-cut and static vibration modes, and on the other a high amplitude blow that brings both heavy pitch glides and chaotic oscillations over a large period of time (the much sought after “whooshing” sound). This example also pinpoints a creative perspective of this kind of modelling: in real life, it may be impossible to strike a stiff beam with such force that it enters into a chaotic regime. However there is no problem in doing so here. It also means

that these rich behaviours can be obtained for almost any given model topology, which is (to say the least) an exciting perspective for sound exploration.

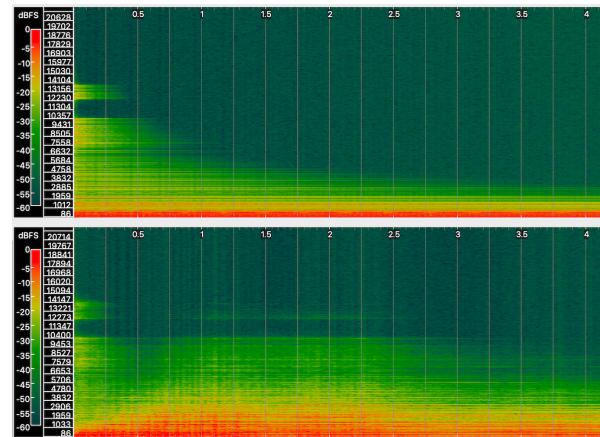


Figure 11. Emerging effect of chaos in the beam model shown in fig. 10. Top: low amplitude excitation. Bottom: high amplitude excitation.

These preliminary observations confirm that generalised non-linear behaviour of physical matter - the current *hot topic* of physically-based sound synthesis - is present by essence in multi-dimensional MI virtual objects. Bearing in mind the potential of such behaviours in the context of “creative” modelling, we believe that mass-interaction physics is a highly relevant method with a key role to play in formally understanding, designing and manipulating virtual vibrating objects that exhibit non-linearities.

## 7. CONCLUSION AND PERSPECTIVES

This entire paper has been dedicated to highlighting the potential the authors foresee about 3D mass-interaction models for sound and music creation. The methodology in itself is worth mentioning, as a necessary step was to step back from sound-synthesis in order to consider environments and tools that include, as a *continuous* and *fully integrated* workflow, every key aspect for exploring the potential of multisensory 3D mass-interaction physical models: real-time computation and rendering of 3D scenes, sound synthesis capabilities, both control and haptic interaction, a language that enables scripting for model design and setting, etc.

As a significant outcome of this prototyping tool, experimental validation proves that within the scope of sound synthesis, such models naturally yield emergent non-linear physical behaviour and - further still - do so without any added mathematical or modelling complexity.

More generally, it appears that the scalar world, in which are built most audio-based DSP environments, might not be the most suited to elaborate new paradigms for musical creation based on 3D mass-interaction physical models. **Multi-dimensional geometry** is the key. And not simply as a consideration helping to create meshes whose acoustic behaviour can then reduced to 1D or 2D, but as a necessary level of description and calculation of virtual physical matter.



From this point onward, all is yet to be done regarding :

- Strengthening the genericity of miPhysics in terms of languages and environments. Processing is an invaluable prototyping tool but not an end in itself.
- Positioning fully miPhysics considering the literature of human computer interaction, computer music and specifically computer graphics previous [20] and recent [21] problematization and results.
- The exhaustive formalisation and characterisation of MI multi-dimensional modelling, both through analytical considerations and empirical studies.
- Reflecting upon how one listens to multi-dimensional and spatially distributed virtual physical objects.
- Designing model analysis tools for such objects by extending the topology-based modal analysis approach.
- Taking the geometrical aspect further: all contacts are currently defined as *sphere-to-sphere* interactions between two punctual material elements. Geometrical surface modelling and contact handling from CGI and haptics should allow to extend the formalism.

As a general conclusion, we affirm that mass-interaction physics is still a potent framework for sound-synthesis, and that it should not be put on a shelf as a part of physical modelling history just yet... but don't take out word for it - grab the library<sup>14</sup> and get coding to see for yourself!

## 8. REFERENCES

- [1] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. Chichester, UK: John Wiley and Sons, 2009.
- [2] J. O. Smith, *Physical Audio Signal Processing: for Virtual Musical Instruments and Digital Audio Effects*. W3K Publishing, 2010.
- [3] —, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, no. 4, pp. 74–91, Winter 1992.
- [4] J.-M. Adrien, "The missing link: Modal synthesis," in *Representations of musical signals*. MIT Press, 1991, pp. 269–298.
- [5] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen, "Discrete-time modelling of musical instruments," *Reports on progress in physics*, vol. 69, no. 1, p. 1, 2005.
- [6] C. Cadoz, A. Luciani, and J. L. Florens, "Cordis-anima: a modeling and simulation system for sound and image synthesis: the general formalism," *Computer music journal*, vol. 17, no. 1, pp. 19–29, 1993.
- [7] J. Leonard and C. Cadoz, "Physical modelling concepts for a collection of multisensory virtual musical instruments," in *New Interfaces for Musical Expression 2015*, 2015, pp. 150–155.
- [8] C. Roads, "Interview with max mathews," *Computer Music Journal*, vol. 4, no. 4, pp. 15–22, 1980.
- [9] S. Bilbao, "The changing picture of nonlinearity in musical instruments: Modeling and simulation," in *Proc. Int. Symp. Musical Acoustics*, 2014.
- [10] N. Castagné and C. Cadoz, "Genesis: a friendly musician-oriented environment for mass-interaction physical modeling," in *ICMC 2002-International Computer Music Conference*, 2002, pp. 330–337.
- [11] S. Rimell, D. M. Howard, A. M. Tyrrell, R. Kirk, and A. Hunt, "Cymatic. restoring the physical manifestation of digital sound using haptic interfaces to control a new computer based musical instrument," in *ICMC*, 2002.
- [12] M. Evrard, A. Luciani, and N. Castagné, "Mimesis: Interactive interface for mass-interaction modeling," in *International Conference on Computer Animation and Social Agents*, 2006, pp. 177–186.
- [13] K. Sillam, M. Evrard, and A. Luciani, "A real-time implementation of the dynamic particle coating method on a gpu architecture," in *4th Workshop in Virtual Reality Interactions and Physical Simulation 2007*. Eurographics Association, 2007, pp. 69–78.
- [14] J. Leonard and J. Villeneuve, "Fast audio-haptic prototyping with mass-interaction physics," in *International Workshop on Haptic and Audio Interaction Design (HAID'19)*, 2019.
- [15] J.-C. Risset, "Computer study of trumpet tones," *The Journal of the Acoustical Society of America*, vol. 38, no. 5, pp. 912–912, 1965.
- [16] D. Roze and J. Bensoam, "Nonlinear physical models of vibration and sound synthesis," in *Unfold Mechanics for Sounds and Music*, 2014, pp. 1–1.
- [17] V. Valimäki, T. Tolonen, and M. Karjalainen, "Plucked-string synthesis algorithms with tension modulation nonlinearity," in *Conference on Acoustics, Speech, and Signal Processing. Proceedings.*, vol. 2. IEEE, 1999, pp. 977–980.
- [18] J. Villeneuve, C. Cadoz, and J. Leonard, "Analyse de modèles physiques, modèles physiques d'analyse," *Traitement du Signal*, vol. 32, no. 4/2015, pp. 365–390, 2015.
- [19] N. H. Fletcher and T. D. Rossing, *The physics of musical instruments*. Springer Science & Business Media, 2012.
- [20] K. van den Doel, P. G. Kry, and D. K. Pai, "Foleyautomatic: Physically-based sound effects for interactive simulation and animation," *SIGGRAPH '01 Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001.
- [21] J.-H. Wang, A. Qu, T. Langlois, and D. James, "Toward wave-based sound synthesis for computer animation," *ACM Transactions on Graphics*, vol. 37, pp. 1–16, 07 2018.

<sup>14</sup> [www.mi-creative.eu](http://www.mi-creative.eu)

# Exploring the Effects of Diegetic and Non-diegetic Audiovisual Cues on Decision-making in Virtual Reality

Anil Çamcı

University of Michigan

Department of Performing Arts Technology

acamci@umich.edu

## ABSTRACT

The user experience of a virtual reality intrinsically depends upon how the underlying system relays information to the user. Auditory and visual cues that make up the user interface of a VR help users make decisions on how to proceed in a virtual scenario. These interfaces can be diegetic (i.e. presented as part of the VR) or non-diegetic (i.e. presented as an external layer superimposed onto the VR). In this paper, we explore how auditory and visual cues of diegetic and non-diegetic origins affect a user's decision-making process in VR. We present the results of a pilot study, where users are placed into virtual situations and are expected to make choices upon conflicting suggestions as to how to complete a given task. We analyze the quantitative data pertaining to user preferences for modality and diegetic-quality. We also discuss the narrative effects of the cue types based on a follow-up survey conducted with the users.

## 1. INTRODUCTION

Virtual realities are information-rich environments where multiple channels of communication can be formed between the system and the user. Narrative cues in the audio and visual domains are used to guide the users through their experiences in VR. These cues can be presented in the form of user interface elements superimposed onto the VR. They can also be built into the virtual environment itself as objects situated in the implied universe of the VR.

Similar distinctions between the elements of storytelling are used in film, theatre and games in service of diverse narrative goals. For instance, the conversation between the characters in a film can be contradicted by a narrator to warn the audience of a possible deception in the story. Such narrative devices have been utilized in video games with nonlinear gameplay, where users can make decisions upon conflicting situations that lead to branching storylines.

The form in which audio and visual cues are presented to the user in VR can be used for similar effects. In this paper, we explore these effects in the context of a VR game

in which the users are expected to make decisions to exit a virtual room in light of conflicting audio and visual cues that are presented diegetically (within the room) and non-diegetically (as an external layer). We discuss the results of a study where the users were presented with various combinations of such cues. The use of concurrent conflicting cues allows us to investigate within-subject preferences based on modality and diegetic quality. Moreover, we explore the narrative functions that users impart to contradicting cues in terms of their origin and trustworthiness.

In our analysis, we look at decision types and timings, as well as modality and diegetic-quality pairings. Furthermore, we evaluate the qualitative responses gathered from a follow-up survey to highlight the ways in which the users interpreted the various cue types in their decisions and how these interpretations affected their narrative experience.

## 2. RELATED WORK

Modern virtual reality systems are constrained by hardware limitations to a much lesser extent than they were a decade ago. An increasing number of researchers are therefore able to focus on experiential qualities of VR. Accordingly, several design guidelines for VR have been proposed in the recent years [1,2]. We are arguably in the early days of formulating a VR theory akin to that of more established art forms; numerous researchers and practitioners work towards a deeper understanding of how we perceive modern virtual realities, and how we behave in them.

For instance, Naz et al. explore the links between affective qualities of a virtual space and its visual design parameters [3]. The researchers find that the hue and brightness of the colors used in a virtual room impacts the user's affective appraisal of the space in terms of how warm, spacious, intimate or exciting it is perceived to be. Dealing with a similar research question in a multimodal context, McArthur et al. argue that the spatial attention of the user in a virtual environment is coordinated across modalities that encode information differently, and that the prioritization of information in a virtual environment relies on the complex interactions between the individual modalities [4]. Accordingly, the current project adopts a cross-modal approach in its investigation of how the diegetic quality of cues in VR can affect the user experiences.

In 1995, Beroggi et al. hypothesized that VR can be used to support decision-making in emergency management, highlighting the potential of VR for training appli-

Copyright:© 2019 Anil Çamcı. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



Figure 1. An isometric view of the virtual room where the user study was conducted with its surrounding walls removed for this image. The user is tasked with picking a key and choosing a door to exit the room; while doing so, they are presented with conflicting suggestions in the form of diegetic and non-diegetic audio and visual cues.

cations [5]. Today, VR is viewed as a suitable platform for conducting behavioral studies including those that pertain to psychophysical operations [6,7] and decision-making [8–10]. Whereas previous projects have utilized VR simulations to study decision-making processes in life-critical situations [11,12], the current project adopts a user-experience approach to the study of decision-making in VR.

The distinction between diegetic and non-diegetic sounds are already used in games as narrative devices and means of diversifying the modes of interaction for the user [13]. Non-diegetic sounds, for instance, are often used to cue the players into certain actions, or alert them of state changes [14]. Summers and Jesse argue that the use of diegetic and non-diegetic sounds in VR are integral to conveying elements of narrative [15].

In their study of similar factors in the visual domain, Salomoni et al. suggest that the appropriate use of diegetic and non-diegetic visual interfaces in VR can have a significant impact on the sense of presence in immersive simulations [16]. Accordingly, Nielsen et al. identify the diegetic nature of a cue as a primary dimension in their taxonomy of cues for guiding the user's attention in VR [17].

Exploring methods of guiding user attention in cinematic VR, Rothe and Hußmann perform an analysis of users' viewing directions when they are given diegetic sound and lighting cues. The researchers find that objects connected with sounds attract more attention and guide the viewing more effectively [18]. Accordingly, Mateer characterizes the act of locating diegetic sounds in cinematic VR as a "natural tendency" [19].

In a study that explores the effects of diegetic and non-

diegetic user interface elements on game immersion, Iacovides et al. find that while removal of non-diegetic interfaces can increase the level of cognitive involvement for expert users, it does not have a notable influence on the experience of novice users [20]. Exploring similar effects in VR, Cliburn and Rilea evaluate the impact of signage on navigation speed in VR. Comparing cases in which the users were able to rely on either diegetic signs or non-diegetic maps shown as head-up displays, the researchers find that subjects who navigate the world based on the signs are significantly faster than those who use maps [21]. These studies indicate how the diegetic qualities of elements presented to the users in simulated experiences can affect performance and engagement.

### 3. DIEGESIS IN VR

The concept of diegesis is applied to a variety of narrative forms, such as film, theatre, and games, to explain how elements of narration are situated in reference to the implied universe of a story [22]. While diegetic elements are those that belong to this universe, non-diegetic elements are external to it.

Diegesis gains further significance in VR studies due to the inherent affordances of the medium, such as constant immersion, dynamic first-person view, and interactivity. Whereas the user assumes an outside perspective towards both diegetic and non-diegetic elements in most narrative forms, VR can situate the user as a diegetic actor in its implied universe. For instance, in film-making, the perspective from which the audience observes a narrative is



pre-determined by the director. In VR, the user can not only shift their perspective but also interact with the narrative [23]. The diegetic disposition of audio and visual elements in a simulation can therefore have an impact on the user's experience based on whether they are internal or external to the virtual space that the user occupies [20].

### 3.1 Visual Diegesis

In VR, the visual objects situated in the virtual space are inherently diegetic. These objects are part of the 3D environment and maintain their positions relative to the world-space of the VR. Diegetic visual objects are often affected by the physical forces implemented in the virtual environment, and interact with other objects accordingly. The lighting and other occlusion effects can alter the visibility of these objects.

On the other hand, non-diegetic visual objects are most commonly presented in the form of visual overlays. These can be head-up displays that relay relevant information about the VR, or user interface (UI) elements such as menus and buttons. Non-diegetic visual elements persist over the user's field of view and are commonly positioned relative to the user rather than the world-space of the VR. These objects are usually unaffected by lighting and occlusion. Some non-diegetic UI elements can be spatially mapped into the virtual environment while remaining external to the implied universe of the VR.

### 3.2 Auditory Diegesis

Similar to visual elements of the same nature, diegetic audio objects in VR belong to the virtual environment. These sounds often originate from visual elements in the scene and are subjected to localization cues based on the user's position. These sounds are also affected by room acoustics and occlusion.

Non-diegetic audio objects in VR are unaffected by the user's position, room acoustics or occlusion effects. Much like non-diegetic visual objects, these are presented in an additional auditory layer that persists over the virtual world. These objects can be alert sounds, voice messages relayed to the user from outside of the virtual space, or backing tracks similar to the non-diegetic score of a film.

### 3.3 Sample Scenario: Park Simulation

In a VR that simulates an outdoor park, examples of diegetic visual objects could be the trees and the benches in the park. A non-diegetic visual object would be the map of the park presented as a head-up display. Whereas the user can look away from a bench in the scene, the map would always remain in the user's field of view.

In the same simulation, an example of a diegetic audio object would be the sound of a bird chirping. Attached to the diegetic visual representation of a bird, this sound would display localization properties relative to the user's position as the bird flies around. When the user reaches their destination at the far end of the park, a non-diegetic bell sound could indicate the successful completion of a

task. This sound would be detached from the visual objects in the scene and heard without localization cues.

## 4. USER STUDY

We conducted a pilot study to evaluate how the modality and diegetic quality of cues in VR can affect the user's decision-making process. The study is designed as a game where the user is expected to make a series of decisions to exit a virtual room in presence of conflicting audio and visual cues of diegetic and non-diegetic nature.

### 4.1 Study Design

The game is designed with Unity and the HTC Vive System as a room-scale VR in a 5m by 5m open space. The virtual room seen in Fig. 1 is mapped onto this space. At one end of the room are two doors. At the other end is a table with two keys on it. The room is lit with a floor lamp placed in one of the corners. A loudspeaker is placed between the two doors above head-level. The purpose of the game is to pick a key and a door to exit the room.

The keys and the keyholes in the doors are the only interactable elements in the scene. The system registers a door selection once a key is inserted in one of the keyholes. When this selection is registered for the first time, the system resets the environment regardless of the selection, prompting the user to repeat the task. Once the system registers a door selection in the second attempt, the room is removed from the scene, effectively placing the user outside. This indicates that the user has successfully completed the task. We implemented the 2-attempt model to alleviate the effects of random decisions by encouraging the users to strategize over their decisions, and to monitor how users evaluate modal and diegetic pairings once a choice combination is perceived to be inaccurate.

Various combinations of audio and visual cues are presented in different configurations of the game. These cues include:

- **Diegetic visual (DV) cues:** a note on the table between the two keys, suggesting a key to pick, and a poster on the wall between the two doors, suggesting a door to choose;
- **Non-diegetic visual (NDV) cues:** head-up display messages that appear at the bottom of the screen when the user enters the collider around the table suggesting a key to pick, or when the user enters the collider around the doors, suggesting a door to choose;
- **Diegetic audio (DA) cues:** periodic announcements made through the loudspeaker in the room, suggesting a key to pick or a door to choose accompanied by classical music;
- **Non-diegetic audio (NDA) cues:** voice messages that are played-back when the user enters the collider around the table suggesting a key to pick, or when the user enters the collider around the doors, suggesting a door to choose.



The diegetic audio cues are spatialized binaurally using Google's Resonance Audio SDK for Unity. The cues are therefore subjected to room and distance effects, and are spatially mapped to the virtual speaker in the room. To better emulate the output of a loudspeaker, these sounds are given directionality characteristics so that they are heard more clearly when the user is within the dispersion field of the speaker. The cues are accompanied by classical music to give the user a constant sense of localized sound at the times the announcement is not being repeated. The music is ducked (i.e. attenuated) with a side-chain compressor when an announcement is being made in style of radio announcements. In configurations of the study where diegetic audio cues for both the keys and the doors are offered, the cues are compounded into a single announcement (e.g., "Pick the purple key and choose the door on the left").

The non-diegetic audio cues are played back without any localization or room effects. They are therefore detached from visual sources in the virtual room, and are perceived as originating from the user's current position. These cues are triggered each time the user enters one of the colliders surrounding the table or the doors. Both diegetic and non-diegetic audio cues are spoken by a neutral female voice.

Timestamps are generated when the user enters and exist the colliders surrounding the table and the doors. The times at which the user picks up a key, and inserts the key in a keyhole are also tracked. Additionally, the global time at which the game is started, and the time at which the game is reset for the second attempt are stored.

#### 4.1.1 Scenarios

The study consists of 12 scenarios based on the different combinations of audio and visual diegetic and non-diegetic cues. These combinations are shown in Table 1. Between scenarios 1 and 2, 5 and 6, and 9 and 10, the cue types are swapped between the key and the door suggestions. In scenarios 3, 4, 7, 8, 11 and 12, the cue types are maintained across the two suggestions to control for within-modality and within-diegetic-quality conditions. Between scenarios 3 and 4, and 7 and 8, the diegetic quality of the cues are swapped while modalities are maintained, whereas between 11 and 12, the modalities of are swapped while diegetic qualities are maintained.

## 4.2 Participants

24 participants (15 male, 9 female; mean age: 23) took part in the current study. 12 participants described themselves as experienced VR users. 5 participants reported having tried VR before, while 7 indicated that this was their first time trying VR. The participants were evenly distributed across the 12 scenarios listed in Table 1.

## 4.3 Procedure

The study takes approximately 15 minutes to complete with 3 minutes for instructions and preparations, 2-3 minutes for the VR portion, and 10 minutes for the follow-up survey. In the instructions, the following items are communicated to the user:

	Purple Key	Green Key	Left Door	Right Door
1	DA	NDA	DV	NDV
2	DV	NDV	DA	NDA
3	DA	DV	DV	DA
4	NDA	NDV	NDA	NDV
5	DA	NDV	NDA	DV
6	DV	NDA	DA	NDV
7	DV	NDA	NDA	DV
8	NDV	DA	DA	NDV
9	NDV	NDA	DA	DV
10	DV	DA	NDA	NDV
11	NDV	DV	DV	NDV
12	DA	NDA	NDA	DA

Table 1. Distribution of diegetic audio (DA), non-diegetic audio (NDA), diegetic visual (DV), and non-diegetic visual (NDV) cues in each of the 12 scenarios. Each participant is placed into one of these scenarios.

- You will be placed in a virtual room that is mapped to the physical space of the study area. In this room, you will find two keys and two doors: only one of the keys will open only one of the doors, and you will be given two attempts to find the right combination to exit the room. There are no hidden clues or puzzles that can guide you out.
- You might, however, encounter various combinations of audio and visual cues that guide you in your decisions; these cues might conflict with each other and it will eventually be up to you to make a decision. When you insert a key in a keyhole, the system will register this as a decision.

The use of the Vive Controller to interact with the keys is demonstrated to the user. The user is then asked to take a seat in the designated chair, and put on the head-mounted display. The room-scale VR experience is initiated with the user seated in the virtual room configured for one of the 12 scenarios. The virtual chair is precisely aligned with the one in the real world. Once in VR, the user can explore the room, and make decisions relevant to the task without a time constraint. When the first pair of decisions is made, the game gets reset with the same cues in place, indicating to the user that they did not make it out of the room. When the second pair of decisions is made, the user is placed outside, indicating that the game is successfully completed.

As a follow up to the VR portion of the study, the user is invited to respond to a qualitative survey in an interview format. In this survey, they are asked to identify the audio and visual cues they encountered in the scene, where they thought these cues might have originated from, and whether the cues were correlated in any way. They are then asked to describe their thought process in making each of their decisions in VR.

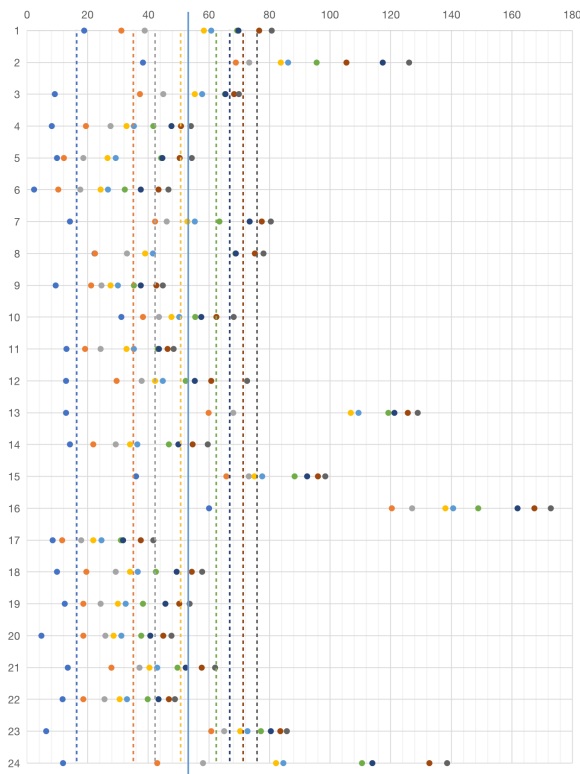


Figure 2. Action timings for each participant with the Y-axis indicating participant number and the X-axis indicating time in seconds. The consecutive dots from left to right indicate the times for: entry to the collider around the table, key pick-up, entry to the collider around the doors, key insertion in a keyhole, scene restart, second entry to the collider around the table, second key pick-up, second entry to the collider around the doors, second key insertion in a keyhole. The dashed lines indicate average times for each of these actions with the average scene restart time indicated with a solid blue line.

#### 4.4 Results and Discussion

The users described their overall experience as responsive and convincing. None of the users reported having experienced discomfort or hard time interacting with the system. During the post-study interview, all users correctly identified the cues they encountered in VR.

The timings of the actions performed by each user are shown in Fig. 2. The average time to complete the study was 75.8 seconds ( $SD = 34.2$ ). First-time users have completed the study in 56 seconds on average ( $SD = 8.7$ ). Those with prior experience with VR completed the study in 84 seconds on average ( $SD = 37.6$ ). The average action times shown in Fig. 2 indicate that the users spent the most amount of time during the initial exploration of the VR, and while making the decision to pick up a key for the first time. The increased speed in choosing a door observed with some users can be attributed to the fact that certain scenarios expose the users to diegetic cues about doors as soon as the game is started.

In repeated conditions (i.e. scenarios 3, 4, 7, 8, 11, 12

	<b>DA</b>	<b>NDA</b>	<b>DV</b>	<b>NDV</b>
Key Decision	25%	75%	75%	25%
Door Decision	25%	75%	25%	75%
	<b>DA</b>	<b>DV</b>	<b>NDA</b>	<b>NDV</b>
Key Decision	25%	75%	50%	50%
Door Decision	75%	25%	50%	50%
	<b>DA</b>	<b>NDV</b>	<b>DV</b>	<b>NDA</b>
Key Decision	100%	0%	50%	50%
Door Decision	100%	0%	75%	25%

Table 2. Frequency distribution (%) of decisions made in the first attempts for each conflicting cue pairs where 25% represents a decision by 1 participant.

seen in Table 1), 9 out of 12 users followed the same cue type (e.g., DA in Scenario 3) for the first set of decisions, and followed the other cue type (e.g., DV in Scenario 3) in their second attempt. All 3 users who followed mixed cue types in their first attempt (e.g., DA and DV in Scenario 3) indicated that they thought the system was trying to deceive them. While one of these users followed a matching cue type in their second attempt, the other 2 users continued to follow mixed cue types in their second attempts. These users expressed that even though they imagined different instigators behind the audio and visual cues, they thought all of these characters were trying to trick them.

Some users interpreted the diegetic visuals as clues left behind by game characters who had previously gotten out of the room, and the non-diegetic visuals as originating from an authority figure who designed the game. Other users have also expressed that the sense of an external authority was a decisive factor.

Table 2 shows the decision rates for each of the 6 cue combinations (i.e. conflicting concurrent cues) that were presented in relation to the key and the door decisions. A consistent preference is not observed in the conditions where diegetic qualities are matched (i.e. DA-DV, and NDA-NDV). In the audio cue combinations (i.e. DA-NDA), a preference towards non-diegetic cues can be seen. A strong preference towards diegetic audio cues is observed in the DA-NDV combination.

Once the initial decisions were found out to be wrong after the first attempt, the users followed a variety of approaches in their second attempts: 7 users chose the same key but a different door, 6 chose a different key but the same door, and 11 changed both of their decisions. Some of these decisions were reported to be based on maintaining a diegetic consistency in the first attempt and a modal consistency in the second. Table 3 shows the instances where the users have maintained a pairing in terms of modality or diegetic quality between their decisions about the key and the door. For instance, User 18 followed the diegetic (i.e. DV and DA) cues for both decisions in their first attempt, while they followed the visual (i.e. DV and NDV) cues in their second attempt. Overall, diegetic-quality pairings were more common in the first attempts, whereas visual pairings were a prominent choice in the second attempts.

The users' perception of whether the cues were addressed

Scenario	1 <sup>st</sup> Attempt				2 <sup>nd</sup> Attempt			
DA-NDA; DV-NDV	1	-	13	ND	1	D	13	-
DV-NDV; DA-NDA	2	-	14	-	2	ND	14	-
DA-DV; DV-DA	3	-	15	-	3	V	15	-
NDA-NDV; NDA-NDV	4	V	16	A	4	-	16	V
DA-NDV; NDA-DV	5	D	17	D	5	V	17	V
DV-NDA; DA-NDV	6	A	18	D	6	V	18	V
DV-NDA; NDA-DV	7	DV	19	ND	7	NDA	19	-
NDV-DA; DA-NDV	8	DA	20	NDA	8	-	20	-
NDV-NDA; DA-DV	9	V	21	-	9	A	21	V
DV-DA; NDA-NDV	10	-	22	-	10	A	22	V
NDV-DV; DV-NDV	11	-	23	D	11	-	23	ND
DA-NDA; NDA-DA	12	-	24	D	12	D	24	ND

Table 3. Modality and diegetic-quality pairings formed by the participants between their key and door choices in the first and second attempts. Scenarios highlighted in gray consist of repeated conditions where modality or diegetic-quality pairings may not be possible in the first place.

to them privately or in the form of public messages had an impact on how they evaluated the suggestions. For instance, a user placed in a DA-DV scenario thought the “silent” note on the table was more trustworthy than the announcement that others could potentially hear. Conversely, in every condition where an NDV cue was presented in conflict with a DA cue, the users preferred the latter, stating that the NDV cue seemed “robotic”, “inhuman” and “unidentified”.

Furthermore, NDV was the least preferred cue type for the first key choice overall with only 3 users following NDV cues for this choice. One of these users expressed having trusted the DV cue first but decided that the NDV cue felt like a more recently updated source. Another user reported not having paid attention to the NDV cue in their decision. The third user mentioned having decided to go with a modal parity; therefore, the DV cue about the doors motivated their preference towards the NDV cue with the keys.

Some users expressed a preference for diegetic cues, mentioning that these were “actual sources” in the room, whereas non-diegetic sources were “surreal” or “unidentified”. On the other hand, after following a diegetic cue for the key choice, some of these users maintained the modality of this cue in their door choices, even though this meant following a non-diegetic cue.

5 out of 18 users who were placed in a scenario that consisted of a diegetic audio cue expressed uncertainty about whether the diegetic sound was coming from the speaker in the scene. However, these users described the cue with such terms as “intercom”, “announcement”, “PA sound”, and “in the space”, which indicate an association of the sound with the environment. As in most modern applications of binaural audio, a standardized head-related transfer function (HRTF) was used for audio spatialization in the current study. Standardized (i.e. non-individualized) HRTFs can be prone to localization errors due to disparities

between the head model used in the function and the user’s anatomical features [24]. This might explain the cases in which the users were not certain about the spatial correspondence between a diegetic sound and the speaker model in the scene.

There was a notable preference towards NDA cues when presented in conflict with the DA cues. The terms used to describe the NDA cues were “Voice of God”, “narration”, “exogenous”, “in the head” and “disembodied”. These descriptors indicate an interpretation of NDA cues as originating from an external source. While this non-diegetic quality was a cause of disinclination to NDV cues, it served a reassuring role for some users when presented in the audio domain.

## 5. CONCLUSION

VR is transforming our relationship with arts and entertainment: whereas the spectators have often been situated as consumers of such spectacles, VR offers experiences where they can be users and even performers. This distinction brings about many new and interesting challenges for VR researchers and content creators. An increasing number of studies from a wide range of fields, such as human-computer interaction, user experience design, psychology and serious games, are now dealing with such challenges.

The preliminary results gathered from the current study indicate that the diegetic quality of a cue in VR can have a noticeable effect on the decisions made by the users when presented with conflicting cues to complete a task. We believe that these results can inform the design of interactive VR experiences, such as those afforded by games and VR films.

While some users followed modal parities in their decisions, others paired cues in terms of diegetic quality. A strong preference for diegetic audio cues over non-diegetic visual cues was observed. Another notable preference was towards non-diegetic audio cues over diegetic audio cues. Despite these correlations, a preference was not observed between non-diegetic cues of different modalities.

We plan to expand our sample size in future iterations of the study in order to improve the statistical significance of our results. We also plan to investigate the narrative function of cues in isolation rather than in conflicting situations to better understand between-subject preferences towards modality and diegetic qualities. When paired with the qualitative feedback from the users, the current results show that the diegetic quality of a cue can serve as a conceptual indicator that affects the user’s narrative interpretation of a VR experience and how they group elements of the VR. This affect was observed when users commonly associated the cues with such concepts as trust, authority, deception, and privacy.

We hope to further explore such conceptual implications of the modal and diegetic qualities of the elements in VR that can facilitate the design of compelling user experiences. Moreover, we plan to incorporate Witmer and Singer’s presence questionnaire [25] in our survey to control for the effects of such elements on perceived immersion.

## 6. REFERENCES

- [1] R. Yao, T. Heath, A. Davies, T. Forsyth, N. Mitchell, and P. Hoberman, "Oculus VR best practices guide," 2014.
- [2] Google, "Designing for google cardboard," <https://designguidelines.withgoogle.com/cardboard/>, Accessed on 2018-10-28.
- [3] A. Naz, R. Kopper, R. P. McMahan, and M. Nadin, "Emotional qualities of vr space," in *Proceedings of IEEE VR Conference 2017*. IEEE, 2017, pp. 3–11.
- [4] A. McArthur, R. Stewart, and M. Sandler, "Sounds too true to be good: diegetic infidelity—the case for sound in virtual reality," *Journal of Media Practice*, vol. 18, no. 1, pp. 26–40, 2017.
- [5] G. E. Beroggi, L. Waisel, and W. A. Wallace, "Employing virtual reality to support decision making in emergency management," *Safety science*, vol. 20, no. 1, pp. 79–88, 1995.
- [6] R. Skarbez, S. Neyret, F. P. Brooks, M. Slater, and M. C. Whitton, "A psychophysical experiment regarding components of the plausibility illusion," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 4, pp. 1369–1378, 2017.
- [7] T. Ye, S. Qi, J. Kubricht, Y. Zhu, H. Lu, and S.-C. Zhu, "The martian: Examining human physical judgments across virtual gravity fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 4, pp. 1399–1408, 2017.
- [8] M. Minderer, C. D. Harvey, F. Donato, and E. I. Moser, "Neuroscience: virtual reality explored," *Nature*, vol. 533, no. 7603, p. 324, 2016.
- [9] J. Smith, "Immersive virtual environment technology to supplement environmental perception, preference and behavior research: a review with applications," *International journal of environmental research and public health*, vol. 12, no. 9, pp. 11 486–11 505, 2015.
- [10] F. La Paglia, C. La Cascia, R. Rizzo, G. Riva, and D. La Barbera, "Decision making and cognitive behavioral flexibility in a ocd sample: a study in a virtual environment," in *Annual Review of Cybertherapy and Telemedicine 2015: Virtual Reality in Healthcare: Medical Simulation and Experiential Interface*. IOS Press, 2015.
- [11] F. Kaneko, Y. Ikemoto, and M. Fukumoto, "Evacuation simulator for analysis of evacuees' decision in a ship under casualty," in *Proceedings of IEEE VR Conference 1999*. IEEE, 1999, p. 80.
- [12] U. Ju, J. Kang, and C. Wallraven, "Personality differences predict decision-making in an accident situation in virtual driving," in *Proceedings of IEEE VR Conference 2016*. IEEE, 2016, pp. 77–82.
- [13] I. Ekman, "Meaningful noise: Understanding sound effects in computer games," in *Proceedings of Digital Arts and Cultures*, 2005.
- [14] A. Järvinen, "Gran stylissimo: The audiovisual elements and styles in computer and video games," in *Proceedings of Computer Games and Digital Cultures Conference*, 2002, pp. 113–138.
- [15] C. Summers and M. Jesse, "Creating immersive and aesthetic auditory spaces in virtual reality," in *3rd IEEE VR Workshop on Sonic Interactions for Virtual Environments (SIVE) 2017*. IEEE, 2017, pp. 1–6.
- [16] P. Salomoni, C. Prandi, M. Roccetti, L. Casanova, L. Marchetti, and G. Marfia, "Diegetic user interfaces for virtual environments with hmds: a user experience study with oculus rift," *Journal on Multimodal User Interfaces*, vol. 11, no. 2, pp. 173–184, 2017.
- [17] L. T. Nielsen, M. B. Møller, S. D. Hartmeyer, T. Ljung, N. C. Nilsson, R. Nordahl, and S. Serafin, "Missing the point: an exploration of how to guide users' attention during cinematic virtual reality," in *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*. ACM, 2016, pp. 229–232.
- [18] S. Rothe and H. Hußmann, "Guiding the viewer in cinematic virtual reality by diegetic cues," in *Proceedings of International Conference on Augmented Reality, Virtual Reality and Computer Graphics*. Springer, 2018, pp. 101–117.
- [19] J. Mateer, "Directing for cinematic virtual reality: how the traditional film directors craft applies to immersive environments and notions of presence," *Journal of Media Practice*, vol. 18, no. 1, pp. 14–25, 2017.
- [20] I. Iacovides, A. Cox, R. Kennedy, P. Cairns, and C. Jennett, "Removing the hud: the impact of non-diegetic game elements and expertise on player involvement," in *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*. ACM, 2015, pp. 13–22.
- [21] D. C. Cliburn and S. L. Rilea, "Showing users the way: Signs in virtual worlds," in *Proceedings of IEEE VR Conference 2008*. IEEE, 2008, pp. 129–132.
- [22] G. Genette, *Narrative discourse: An essay in method*. Cornell University Press, 1983.
- [23] R. Aylett and S. Louchart, "Towards a narrative theory of virtual reality," *Virtual Reality*, vol. 7, no. 1, pp. 2–9, 2003.
- [24] S. Zhao, R. Rogowski, R. Johnson, and D. L. Jones, "3d binaural audio capture and reproduction using a miniature microphone array," in *Proceedings of Digital Audio Effects (DAFx) Conference 2012*, 2012, pp. 151–154.
- [25] B. G. Witmer and M. J. Singer, "Measuring presence in virtual environments: A presence questionnaire," *Presence*, vol. 7, no. 3, pp. 225–240, 1998.



# OSC-XR: A Toolkit for Extended Reality Immersive Music Interfaces

**David Johnson**  
Department of Computer Science  
University of Victoria  
davidjo@uvic.ca

**Daniela Damian**  
Department of Computer Science  
University of Victoria  
danielad@uvic.ca

**George Tzanetakis**  
Department of Computer Science  
University of Victoria  
gtzan@ieee.org

## ABSTRACT

Currently, developing immersive music environments for extended reality (XR) can be a tedious process requiring designers to build 3D audio controllers from scratch. OSC-XR is a toolkit for Unity intended to speed up this process through rapid prototyping, enabling research in this emerging field. Designed with multi-touch OSC controllers in mind, OSC-XR simplifies the process of designing immersive music environments by providing prebuilt OSC controllers and Unity scripts for designing custom ones. In this work, we describe the toolkit's infrastructure and perform an evaluation of the controllers to validate the generated control data. In addition to OSC-XR, we present UnityOscLib, a simplified OSC library for Unity utilized by OSC-XR. We implemented three use cases, using OSC-XR, to inform its design and demonstrate its capabilities. The Sonic Playground is an immersive environment for controlling audio patches. Hyperemin is an XR hyperinstrument environment in which we augment a physical theremin with OSC-XR controllers for real-time control of audio processing. Lastly, we add OSC-XR controllers to an immersive T-SNE visualization of music genre data for enhanced exploration and sonification of the data. Through these use cases, we explore and discuss the affordances of OSC-XR and immersive music interfaces.

## 1. INTRODUCTION

In 1992 Jaron Lanier performed *The Sound of One Hand*, a live improvisation using the three instruments designed for the EyePhone (an early virtual reality headset) [1]. A remarkable aspect of his performance (aside from the technologies) was that Lanier was able to simultaneously play multiple instruments to perform music that could not easily have been performed with traditional instruments. Lanier's work showed the potential for immersive musical performances, but since then there has been limited research exploring the musical interactions afforded by virtual reality (VR) and related extended reality (XR) technologies. When Serafin et al. [2] recently surveyed the state of art in virtual reality music instruments (VRMIs) in 2016, the number of interfaces available was fairly small. The capabilities and relatively few design constraints of XR create

the potential for a wide array of immersive interfaces based on any of Miranda and Wanderley's four categories of music interfaces: *Augmented Musical Instruments*, *Instrument Like Controllers*, *Instrument Inspired Controllers*, and *Alternate Controllers* [3]. With such broad possibilities, more research is needed to increase our understanding of the affordances of immersive environments and interaction techniques best suited for music control.

To support further research into immersive interfaces for music, we present OSC-XR, a toolkit for rapidly prototyping immersive musical environments in XR using Open Sound Control (OSC), a communication protocol widely used in audio software [4]. Influenced by multi-touch OSC controllers, OSC-XR provides developers with a wide range of readily available components in order to make designing immersive environment more accessible to researchers and sound designers. In this paper, we discuss the infrastructure of OSC-XR, validate its generated data by comparing with a popular multi-touch OSC controller, and present three environments developed to demonstrate its capabilities for immersive interface design.

## 2. RELATED WORK

### 2.1 XR Music Interfaces

In one of the first research studies on virtual music performance, Mulder, Fels and Mase [5] designed virtual 3D instruments that users interacted with using CyberGloves and motion tracking sensors. While the instruments were not displayed in an immersive environment, they did explore interactions in 3D desired for immersive performances. Using a fully immersive environment, Mäki-Patola et al. [6] developed and analyzed four immersive music interfaces based on physical models. In their findings, they reported that because VR is a different medium compared to the real world, mimicking traditional instruments in immersive environments may not result in better instruments unless it is used to augment real instruments with additional control. Rather than mimicking existing instruments, Berthaut et al. [7] proposed 3D reactive widgets for musical performance with interactions that went beyond what is possible in the real world. The reactive widgets represented complex multi-process sounds with many parameters that would be difficult to interact with in the real world. Using VR with carefully designed gestures and audiovisual mappings allowed the user to easily interact with multiple widgets to generate an expressive musical interaction. In these examples, music performance was controlled using

Copyright: © 2019 David Johnson et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

self contained virtual objects tightly coupled with sound generation limiting customization or extensibility.

An immersive interface proposed by Moore et al. [8], called The Wedge, allowed users to not only perform music in VR but also build and customize their performance environment. With this interface a user could build a customized performance environment by selecting and combining note objects from a palette to form musical chords and sequences. The interface used two simple gestures for interaction, one gesture for playing notes and another gesture for building the interface by placing notes within the environment. The interface had limited capabilities for generating sounds as complex as the previously discussed reactive widgets but showed how XR can be used to quickly build customized interfaces for musical performance. Our work takes inspiration from this previous research while also allowing users to build more complex sound environments through customizable OSC controllers.

The previous works all rely on interaction with virtual objects through standard input devices, like hand controllers or data gloves. Bottcher et al. [9] instead proposed a VRMI for interacting with tangible music controllers. In this work, the authors built physical flute and drum like controllers which were represented in the virtual environment as 3D objects. Interaction with the controllers was mapped to the parameters of a physical model. By moving the controllers, the user was able to change the dimensions of the physical model, and it's virtual representation in real-time, while simultaneously using it for a musical control. Using tangible interfaces as controllers for virtual music performance provided users with a clear understanding of the affordances and constraints for interaction.

The presented works provide interesting use cases and examples of immersive environments for musical performance that demonstrate the potential of using XR for musical expression. For a more extensive overview of recent VRMI see the survey on the current state of the art by Serafin et al. [2]. The systems presented, however, are standalone and have limited capabilities for designing new environments. OSC-XR provides designers with a more general toolkit to make building and prototyping new musical environments more accessible.

## 2.2 OSC Controllers

During their research on OSC, Wessel and Wright [4] discussed the affordances of using digitized tablets for musical control as well as potential mapping strategies for gestural control of music. This work has inspired a number of multi-touch OSC based control surfaces. TouchOSC<sup>1</sup> is one of the most popular multi-touch controller applications. It provides users with prebuilt layouts using TouchOSC's standard control widgets. In addition to the set of existing control interfaces, users may also use the TouchOSC Editor to build their own interfaces from the prebuilt widgets. The authors of two other multi-touch toolkits, Argos [10] and Control [11], cited the influence of TouchOSC on their flexible design.

<sup>1</sup> <https://hexler.net/software/touchosc>

Argos was an application for building multi-touch interfaces for musical control using OSC [10]. Using Argos users were able to design control interfaces from a library of prebuilt widgets, such as knobs, sliders and buttons. Additionally, Argos provided developers a set of C++ classes, built on openFrameworks, for creating their own widgets. Similarly, Control, let users design custom interfaces from a set of prebuilt widgets using JSON to define the interface structure [11]. Control was set apart from other interfaces by giving users the ability to add customized functions to their widgets using JavaScript. The popularity of multi-touch OSC controllers, especially TouchOSC, show that OSC based applications with flexible design support needs of designers. While these applications were all designed for multi-touch surfaces, OSC-XR is inspired by the underlying theme of flexible design through prebuilt objects and customized scripting.

Multi-touch devices are not the only place OSC has been used to build control environments. Hamilton [12] used OSC in the design of UDKOSC, a immersive musical performance environment for the Unreal Development Kit (UDK). With this system Hamilton was able to perform in an immersive environment using avatars that interacted with objects in the virtual environment. Our work differs from Hamilton's in that OSC-XR uses a design metaphor based on standard music control idioms rather than the game like metaphor seen in Hamilton's work.

## 2.3 Audio Programming in XR

Immersive environments for XR are typically developed using dedicated game engines, such as Unity<sup>2</sup> or Unreal Engine<sup>3</sup>, which are designed to simplify the process of developing 3D environments through a suite of tools that include advanced graphics rendering pipelines and physics engines. They also include sound engines for playback of sound files with mixing, added effects and sound spatialization. There is, however, minimal support in game engines for audio synthesis capabilities desired by sound designers. Unreal Engine has an experimental package for sound synthesis<sup>4</sup> but the limited features of the environment may not provide sound designers with the full tool set provided existing audio programming languages. To support the design of immersive music environments there is a need for more robust audio synthesis capabilities.

Currently there are some audio programming languages that support audio synthesis and processing with Unity. Faust, for example, can compile to a C library for use as a Unity Plugin [13] and LibPD has a C# wrapper that can be integrated with Unity [14]. Most recently, a plugin to support the use of ChucK within the Unity development environment, called Chunity, was developed [15]. While these systems all add support for audio synthesis to Unity, designers must use Unity scripting to setup parametric control of the patches. Because OSC-XR uses OSC for control, it allows designers to directly integrate their favorite audio synthesis tools into the design process.

<sup>2</sup> <https://unity3d.com>

<sup>3</sup> <https://www.unrealengine.com>

<sup>4</sup> <http://bit.ly/UnrealSynth>

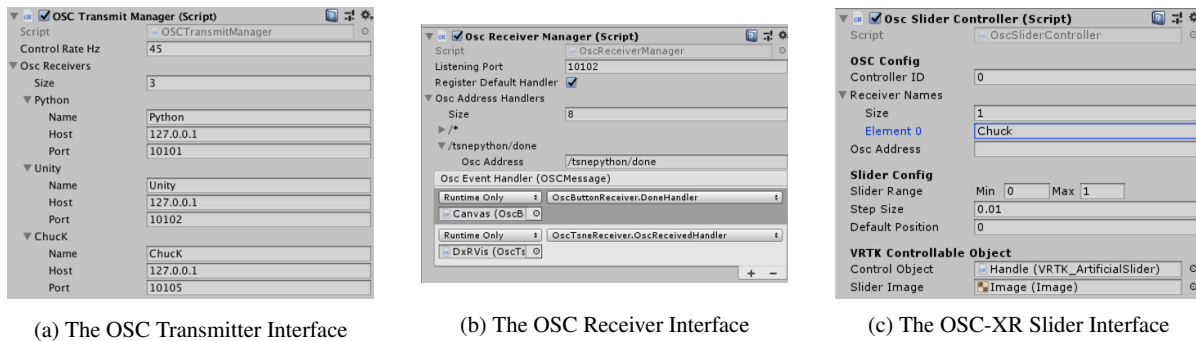


Figure 1: Example Unity Inspector Interfaces for OSC-XR

### 3. UNITY OSC LIBRARY

To implement OSC in Unity, many projects have used Jorge Garcia’s UnityOSC library<sup>5</sup>. We have found this library to be somewhat difficult to integrate in new projects. To simplify the process of OSC configuration we present a new library, UnityOscLib, that builds on Garcia’s core OSC classes and integrates them into the Unity development work flow. The new library simplifies configuration by implementing separate *MonoBehaviour* (a Unity base class from which all Unity scripts must be derived) classes for receiving and transmitting OSC messages. The configuration process has also been simplified by exposing OSC properties in the Unity Inspector as well as through Unity scripting. At the time of writing, we have not exposed OSC bundles or timestamps through the UnityOscLib API (but plan to do so in a later release). This section briefly introduces the new library while complete details, including examples, can be found on the project’s Github repository<sup>6</sup>.

The *OscTransmitManager* is a Unity *MonoBehaviour* that handles all aspects of transmitting OSC messages. To send OSC messages from a Unity application, add the OSC transmit manager to one *GameObject* and configure connection information for one or more OSC receivers. OSC receiver configuration details are exposed through the Unity Inspector, as shown in Fig. 1a, in addition to the scripting interface using the *AddReceiver* method. Once configured, the environment is ready to transmit OSC messages using *SendMessage* or *SendMessageAll*.

The OSC transmit manager also implements an optional control rate feature, to configure the frequency of OSC message transmission. Transmitting OSC messages may be triggered by specific events that only occur periodically, such as collision events, but they may also be triggered continuously, for example when an object’s position is changing. This type of continuous data is generally calculated at a rate specified by Unity’s *Update* or *FixedUpdate* messages. With XR these messages typically occur at around 90 frames per second (FPS) or faster as technologies improve. Audio applications may not be able to handle incoming messages at this rate. The UnityOscLib control rate feature is implemented to limit the rate OSC messages are transmitted. To use this feature, developers should reg-

ister a method that transmits OSC messages with the *OnSendOsc* event of the *OscTransmitManager*. Any methods registered with *OnSendOsc* will be called at the control rate specified in the Unity Inspector.

The *OscReceiverManager* is a Unity *MonoBehaviour* class that manages the routing and handling of incoming OSC messages. To receive OSC messages in a Unity application, add the OSC receiver manager to one *GameObject* in the scene and configure the receiver with the port to listen on, see Fig. 1b. OSC address routing is implemented using Unity Events for configuration in the Unity Inspector as well as using delegate events for C# scripting. To route messages based on in the inspector, UnityOscLib exposes an interface in the inspector to add any number of OSC addresses and one or more handler methods for each address, see Fig. 1b. Additionally, the receiver manager’s *RegisterOscAddress* method is used to add OSC addresses and event handlers through Unity scripting. All OSC event handler methods used should accept a UnityOscLib *OscMessage* as an argument. This implementation provides flexible implementation for adding OSC handling during environment design or at runtime.

### 4. OSC-XR

The main contribution of this work is the OSC-XR toolkit for designing immersive XR environments for music control. It is developed using Unity and UnityOscLib to provide sound designers a simple interface for prototyping interactions in immersive environments. The OSC-XR toolkit contains two main components for building environments, 1) a set of scripts that can be attached to any Unity *GameObject* to transmit the object’s state via OSC and 2) a set of prebuilt music controller, called controller prefabs, for transmitting control data via OSC, similar in concept to widgets in TouchOSC. With this infrastructure, developers with limited Unity experience can quickly design immersive music environments through the use of the controller prefabs. Furthermore, more experienced developers can easily extend custom *GameObjects* with OSC capabilities through the scripting interface. Finally, the robust Unity platform affords customization and extension of any OSC-XR components to those familiar with Unity and C#. The flexible design of OSC-XR, combined with the power of Unity, supports rapid prototyping to make designing im-

<sup>5</sup> <https://github.com/jorgegarcia/UnityOSC>

<sup>6</sup> <https://github.com/fortjohnson/UnityOscLib>

Name	Description	Example OSC Message
<b>OscSlider</b>	A slider prefab with position mapped to a configurable range, see Figs. 1c and 3a	/slider/value 1 4.5
<b>OscPad</b>	A drum prefab with pressed and released events including an optional velocity, see Fig. 3a	/pad/pressed 1 1.5
<b>OscGyro</b>	A virtual gyroscope prefab for sending angular velocities normalized to a range of 0 to 1	/gyro/velocities 1 .9 .7 .5
<b>OscTransform</b>	A script for sending transform data via OSC	/trans/local/pos 1 0.5 1.3 2.0
<b>OscTrigger</b>	A script for sending Unity Trigger events; includes an ID and position information for the triggering object	/trigger/enter 1 0.5 0.4 1.0 2

Table 1: Examples of available OSC-XR controller prefabs and scripts. Refer to our GitHub repository for a complete list.

mersive environments quicker more accessible.

OSC-XR was developed using Unity and tested using the Samsung Odyssey Windows Mixed Reality Headset [16] with SteamVR [17]. By making use of the well known Virtual Reality Toolkit (VRTK) [18], OSC-XR should work with any of VRTK’s supported platforms and hardware, affording multi-platform support. The remainder of this section discusses the OSC-XR infrastructure. The details we provide here are intended to give the reader high level understanding of how the toolkit is structured but we encourage the reader to visit the project’s Github repository<sup>7</sup> for complete details, including video examples.

#### 4.1 OSC Controller Prefabs and Scripts

Adding OSC controller prefabs to a Unity scene is the quickest way to get started with OSC-XR. To implement a controller simply add the prefab from the `OSCXR/Prefabs` folder to the Unity game hierarchy. Once added to the scene, modify the object’s transform as desired. At this point the object is ready to use in the environment. For additional configuration each controller exposes a set of properties in the Unity Inspector, see Fig. 1c. Table 1 lists the descriptions of a few of the available OSC controller prefabs, including an example OSC message for each. Developers can further customize the controller prefabs using Unity tools. For example, the visual aspects of any of the controller prefabs can be modified by configuring the Unity components that comprise each object, such as the meshes or materials.

The OSC-XR scripting interface allows developers to quickly add OSC capabilities to any `GameObject` by attaching any of the readily available controller scripts to the object. Each of the scripts models a predefined behaviour for triggering and sending OSC messages. By default adding an OSC controller script to a `GameObject` uses that object’s state for creating and transmitting OSC messages. This can be overridden on most scripts by updating the `Control Object` property of the script with a different `GameObject`, in which case, the state of the configured `Control Object` will be used instead. This is useful when building a composite object where the tracked object is not the top level object. For example, the slider controller prefab implements this design in which case the state of prefab’s handle is used for control data, as seen

<sup>7</sup> <http://github.com/fortjohnson/OSC-XR>

in Fig. 1c. Table 1 lists the descriptions of a few of the available OSC-XR controller scripts, including an example OSC message for each.

Designers wishing to build their own OSC controller scripts should extend OSC-XR’s `BaseOscController`. This class includes a number of base properties for OSC configuration, the controller ID and the OSC address, as well as methods for sending OSC messages. Furthermore, the class automatically registers the method, `ControlRateUpdate` to support transmitting OSC messages at the control rate specified in the OSC transmit manager. Any controller script that needs to send data at the configured control rate should override `ControlRateUpdate` with a method that generates and transmitting OSC data. Each custom script should extend these options as needed to achieve the behaviour being modeled.

#### 4.2 Control Data Validation

To ensure that data generated by OSC-XR is consistent with users’ expectations, we employ two simple user tasks for comparing OSC-XR with TouchOSC. An OSC receiver is implemented to log data generated by each task for an analysis of user performance. One task utilizes a slider controller (or fader widget in TouchOSC) to validate the control precision of the different applications. The second task utilizes a pad controller (or button widget in TouchOSC) to evaluate rhythmic control of the different interfaces. One of the authors, who has intermediate musical skills, performed the tests to validate that the data sent from OSC-XR is consistent with existing systems.

##### 4.2.1 Slider and Pad Evaluation

To perform the slider evaluation task, a user sets the position of the slider to specific values at regular time intervals. For this work, the task requires setting the values of the sliders to 0.25, 0.75, 0.50, and 1.00 in that order. The user is required to transition the slider to each value on every fourth beat at a tempo of 90 beats per minute (BPM) indicated using a metronome. To perform a baseline analysis of the control data, a user performed the task ten times in both OSC-XR and TouchOSC. The output of the user’s performance is then compared to signal representing the expected data, figures 2a and 2b show the results of each run for both applications overlaid with the expected output. The data is compared quantitatively by calculating



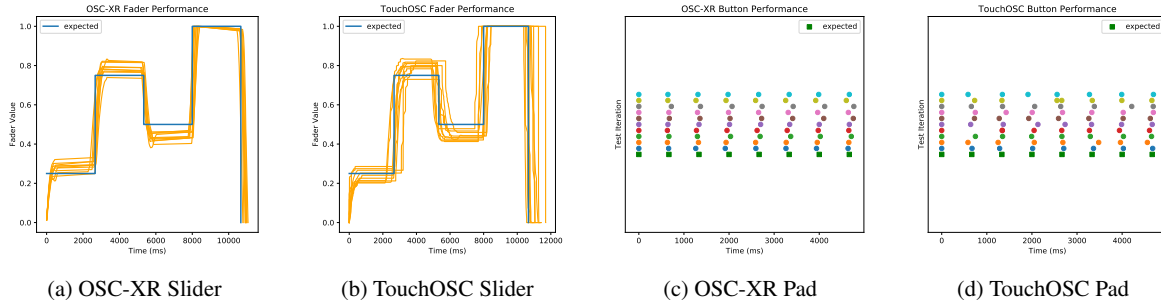


Figure 2: The results of control data validation for slider controllers and pad controllers.

the euclidean distance between the actual and the expected output to calculate an error value. This value is averaged over all iterations for a final error metric. Table 2 lists the average errors for this task for both interfaces.

To perform the pad evaluation task, a user presses a pad controller for eight beats at a tempo of 90 BPM using a metronome to keep time. As in the previous task, baseline analysis of the data is captured with a user that performs the task ten times. Results of each iteration are shown in Figures 2c and 2d, for OSC-XR and TouchOSC respectively. Each iteration is represented as a row of dots where each dot in the row indicates a pad pressed event. For comparison the expected beat times are shown with the green squares in the bottom row. The error for each iteration is calculated as

$$\frac{\sum_{n=1}^N |t_{exp} - t_{act}|}{N} \quad (1)$$

where  $N$  is the number of beats per iteration,  $t_{exp}$  is the expected time of the beat,  $t_{act}$  is the actual time of the pressed event from the user. The errors are averaged over all iterations for the final error metric. The error results for both interfaces are listed in Table 2.

#### 4.2.2 Discussion

Results of the slider evaluation provide a baseline comparison of OSC-XR with TouchOSC. Initial analysis of the data shows similar performance between both applications even though the interactions are slightly different. To move slider in TouchOSC, a user slides their finger across the surface to the new location. Whereas, the OSC-XR slider requires an additional grab interaction to take control of the slider handle before moving it towards its destination. Overall, the OSC-XR slider error is slightly greater than that of TouchOSC. We can compensate for this in OSC-XR by adding a display prefab to the slider for additional feedback. While the interactions required for manipulating sliders are different, this evaluation shows that OSC-XR sliders may perform as well as multi-touch sliders and generate data that is consistent with an application sound designers may already familiar with.

OSC-XR also requires a different technique for interacting with pads due to a lack of haptic feedback. When pressing a pad in OSC-XR users are not provided the same haptic response naturally afforded through interaction with physical objects. Instead users must rely on wrist action

	OSC-XR	TouchOSC
<b>Slider</b>	3.43	2.98
<b>Pad (ms)</b>	35.2	52.0

Table 2: Average errors for each evaluation task

and hand controller momentum to control rhythm. Initial evaluation of the pad controller indicates this may not adversely affect rhythmic performance. Results show that the user was able to perform slightly more accurately with OSC-XR. This may be a result of the user relying on wrist action for control rather than pressing a pad with a single finger. Although a larger study is needed to confirm any hypotheses, users may expect rhythmic control from OSC-XR that is consistent with TouchOSC.

## 5. OSC-XR USE CASES

In this section, we discuss three prototype use cases for immersive environments developed with OSC-XR. Prototyping the environments helped inform the design OSC-XR. Furthermore, the use cases demonstrate the capabilities of the toolkit in different scenarios providing readers ideas on how OSC-XR might be used for their own projects.

### 5.1 The Sonic Playground

The Sonic Playground is an immersive environment that explores a variety of OSC-XR controllers. The playground is composed of multiple zones each with a different performance environment. Users are able to navigate between the zones using teleportation, providing the ability to quickly move between different performance environments. The Sonic Playground is designed to explore and demonstrate musical interaction with OSC-XR controllers that communicate with an external audio programming environment.

The Sampler Zone, seen in Fig. 3a, is an immersive sampler environment composed of a  $3 \times 3$  matrix of pad controllers, to trigger sample playback, and a corresponding matrix of sliders, for additional control of the samples. Pads are configured to send the controller ID as well as pressed and released with included velocity for mapping to sample volume. Each slider is configured to send a value ranging from 0.25 to 5.0 mapped in ChuckK to the playback rate of the corresponding sample. Pad and slider events

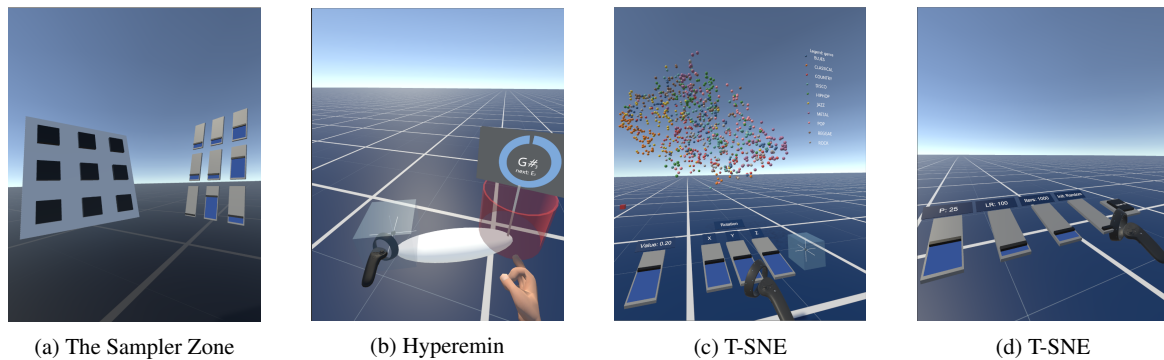


Figure 3: Three immersive environments built using OSC-XR to explore its affordances

are all mapped to a corresponding sample using the controllers' IDs. This environment was developed as a proof-of-concept to demonstrate and explore the affordances of typical music controllers in immersive environments.

The first thing to notice in this environment is the size of the objects. Using input controllers for interaction requires the use of large objects as users lose the dexterity that is naturally afforded through interactions using the hand. Integrating hand tracking devices, such as the Leap Motion, may allow for designing dexterous interactions. Another challenge of performing in XR is the lack of haptic response to physical actions, such as tapping a pad. Even with these challenges, virtual pads in a musical environment afford their own interaction style with large expressive motions and gestures. The evaluation of the pad controller, discussed in Section 4.2, indicates that rhythmic control may not be severely affected through the lack of haptics and in this environment we learned the lack of haptics affords a expressive playing style.

The Sonic Objects zone is an environment for prototyping interactive sound environments. It is composed of various OSC-XR controllers that are readily available to communicate with an audio programming environment, such as ChucK. The environment affords the rapid prototyping of interactive sound design by combining OSC-XR's ability to easily add new controllers and interactions with the power of ChucK's development environment to quickly iterate on sound design.

One of the interesting affordances of immersive music environments we explore is the combination of real life physics based interactions with "impossible" interactions that ignore physics. For example, using physics we can toss objects around or stack and lean them on each other to create interesting soundscapes with generative audio patches. Sometimes, however, a user may want to have more control over when parameters of an audio patch stop as they reach a desired state. By ignoring the physics of an object we can lock it in space to immediately stop it from sending OSC messages. For example, an OscGyro object will always send angular velocity data as its being moved, but a user may want to lock in the sound parameters before releasing the object. With this in mind, we decided to add an interaction to freeze the OscGyro anywhere in space. Once frozen the object will be suspended in space until the user

grabs the object to move it again. Another interesting affordance we discovered through prototyping in this environment is the ability to easily add automation to controllers through Unity components, such as animation or particle systems. For example, the strongly timed behaviour of particle systems allows for particles to collide with an OSC Trigger controller for initiating musical events at rhythmic intervals. Furthermore, the movement of particles within the controller may be mapped to other audio parameters, such as frequency. These examples show how OSC-XR supports rapid prototyping for exploring and creating new musical interaction techniques in XR.

## 5.2 Virtual Hyperinstruments

In the NIME community it is common to augment a traditional instrument with sensors to extend its capabilities. Machover and Chung [19] first presented work on this concept with their hyperinstruments in 1989. Typically hyperinstruments extend traditional instruments with direct augmentation of an instrument, such as a violin, with physical sensors [20]. Physical modification of an instrument can be invasive to its design, therefore, non-invasive techniques have also been developed for augmentation without physical modification, through the use of cameras and depth sensors [21]. These techniques use gesture detection and object tracking for added sound control but provide no visual signifiers to indicate the location of control objects. This is seen in the work of Trail et al. [21] in which they augment a vibraphone with virtual faders that are controlled using mallet tips tracked by a Kinect. Because there are no computer generated signifiers, the fader locations are mapped to the vibraphone keys to signify control locations. Integrating XR in their system would have allowed the authors to add a visual layer to enhance visual feedback.

We have previously explored the virtual hyperinstrument concept by augmenting a physical theremin with virtual objects to visualize the pitch space for music tutoring [22]. We extend that work with Hyperemin, a virtual augmented theremin. OSC-XR controllers are added to the Hyperemin environment to provide real-time control of DSP of the theremin audio. Audio from the theremin is routed to a ChucK patch for playback and audio processing. An OSC 3D Grid controller is added to the environment to control the audio processing, allowing a performer to play

the theremin while also controlling audio processing parameters. Currently, the interaction requires a VR headset and controllers, which may be intrusive to performance but the addition of a LeapMotion sensor, or use of a HoloLens with hand tracking, would address this. In addition to adding sensors directly to the instrument, one of the affordances of XR is the ability to place objects anywhere in the space allowing users to create a customizable control interface not limited to pedals, small device displays or other physical input controllers.

The Hyperemin environment explores the capabilities of OSC-XR for augmenting physical instruments with virtual objects. As XR technology improves we expect that augmenting more traditional instruments will become more accessible. For example, with proper tracking technology it would be possible to attach an OSC Gyro object to the head of a violin and a set of pads to the body adding additional control without physical modifications.

### 5.3 Immersive Vis Control

OSC-XR was designed with music interfaces in mind but its support for rapid prototyping makes it ideal for prototyping other types of immersive environments that require parametric control and distributed communication. With the emergence of XR technologies, there has been trend of research towards immersive environments for information visualization [23]. With this comes the need to rapidly prototype interaction techniques to support the design of immersive interfaces. In this case, we explore the process of prototyping with OSC-XR to build immersive visualization environment with sonic interaction and distributed communication.

To explore interaction needs of immersive analytics environments, we implemented a 3D visualization of the GTZAN music genre dataset [24]. To visualize the high dimensional data in 3D, 52 spectral and timbral features of each song in the dataset are transformed into 3D coordinates using T-SNE [25]. To visualize the data, we integrate OSC-XR with an immersive visualization toolkit, DxR [26]. Using DxR we were able to quickly develop an immersive scatterplot visualization of the T-SNE data. While DxR provides a 3D interface for controlling the visualization, it is limited to basic point and touch based interactions. Integrating OSC-XR into this environment allows us to quickly prototype new interfaces and interactions to control the visualization as well as augment it with additional functionality.

We prototyped a new interface to manipulate the DxR generated visualization by augmenting the environment with new control interfaces and interactions using OSC-XR objects. The interface is composed of two control panels, the main panel is to manipulate the view of the visualization, as shown in Fig. 3c, and a second panel controls the T-SNE parameters, which was not previously possible using DxR alone, shown in Fig. 3d. The main panel provides users a set of sliders to directly manipulate view parameters such as zoom and rotation. Since this panel affects the visualization in real-time and would be frequently utilized by a user during data analysis, it is oriented such that a user

is facing the visualization while interacting with the controller. The T-SNE control panel, oriented to the left of the user, allows users to adjust T-SNE parameters and rerun the data transformation on a Python server without having to leave the virtual environment. We also take advantage of OSC-XR capabilities to interact with the visualization marks from a distance. Every mark in the visualization is configured as an OSC Pointer Trigger providing users the ability to interact with marks using the pointer from an input controller. Using this interaction technique a user is able to select any mark in the visualization to playback its associated audio file allowing users to explore the data aurally, as well as visually. Lastly, visualization marks can be filtered using the pointer by selecting a genre mark from the legend. Using OSC-XR controllers we have been able to quickly prototype new methods for exploring and interacting with an immersive visualization.

While Unity, DxR, and OSC-XR are all used to build the immersive environment, other applications are needed to support it. T-SNE is implemented in Python and audio playback is implemented in Chuck. OSC communication affords us the ability to easily communicate between the distributed applications. In addition, OSC-XR also allows for communication within Unity by attaching OSC receiver methods to Unity `GameObjects` affording flexible and extensible event handling. By using OSC-XR, we are able to rapidly prototype an immersive environment with complex needs, such as toolkit integration and distributed communication.

## 6. CONCLUSION

We have introduced OSC-XR, a toolkit for prototyping immersive musical environments. By providing developers readily available controllers and scripts enabled with OSC, OSC-XR reduces the need to build control objects from scratch, making the development of immersive environments more accessible to researchers and developers. Combined with the power of Unity for building 3D environments, developers using OSC-XR are able to easily explore the affordances of immersive XR environments to find interactions for music control that would not be possible with other mediums.

The flexibility of OSC-XR creates many opportunities to further research on immersive music environments. First, we plan to implement features to spawn any controller prefab from within an immersive environment. This provides sound designers, with and without Unity development experience, the ability to build and customize immersive performance environments on the fly. Furthermore, to allow designers to take full advantage of the large amounts of data potentially created by such an environment OSC-XR would benefit from gesture learning capabilities, similar to those of the Wekinator [27]. Adding these features to OSC-XR will expand the possibilities of immersive performance environments and make designing them more accessible.

## 7. REFERENCES

- [1] J. Lanier. (2019) Virtual instrumentation. [Online]. Available: <http://jaronlanier.com/instruments.html>
- [2] S. Serafin, C. Erkut, J. Kojs, N. C. Nilsson, and R. Nordahl, "Virtual reality musical instruments: State of the art, design principles, and future directions," *Computer Music Journal*, vol. 41, no. 2, 2016.
- [3] E. R. Miranda and M. Wanderley, *New Digital Musical Instruments: Control And Interaction Beyond the Keyboard*. A-R Editions, Inc., 2006.
- [4] D. Wessel and M. Wright, "Problems and Prospects for Intimate Musical Control of Computers," *Computer Music Journal*, vol. 26, no. 3, pp. 11–22, 2002.
- [5] A. G. E. Mulder, S. S. Fels, and K. Mase, "Design of virtual 3d instruments for musical interaction," in *Proceedings of the 1999 Conference on Graphics Interface*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 76–83.
- [6] T. Mäki-Patola, J. Laitinen, A. Kanerva, and T. Takala, "Experiments with virtual reality instruments," in *Proceedings of the 2005 Conference on New Interfaces for Musical Expression*, 2005, pp. 11–16.
- [7] D. F. Berthaut, M. Desainte-Catherine, and M. Hachet, "Interacting with 3d reactive widgets for musical performance," *Journal of New Music Research*, vol. 40, no. 3, pp. 253–263, 2011.
- [8] A. G. Moore, M. J. Howell, A. W. Stiles, N. S. Herrera, and R. P. McMahan, "Wedge: A musical interface for building and playing composition-appropriate immersive environments," in *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, March 2015, pp. 205–206.
- [9] N. Böttcher, S. Gelineck, L. Martinussen, and S. Serafin, "Virtual reality instruments capable of changing physical dimensions in real-time," *Proceedings of Enactive 2005*, 2005.
- [10] D. Diakopoulos and A. Kapur, "Argos: An Open Source Application for Building Multi-Touch Musical Interfaces," in *Proceedings of the 2010 International Computer Music Conference*, 2010, pp. 88–91.
- [11] C. Roberts, "Control: Software for end-user interface programming and interactive performance," in *Proceedings of the 2011 International Computer Music Conference*, 2011, pp. 425–428.
- [12] R. Hamilton, "UDKOSC: An immersive musical environment," in *Proceedings of the 2011 International Computer Music Conference*, 2011, pp. 717–720.
- [13] Y. Orlarey, D. Fober, and S. Letz, "Faust: an efficient functional approach to dsp programming," *New Computational Paradigms for Computer Music*, vol. 290, p. 14, 2009.
- [14] P. Brinkmann, C. McCormick, P. Kirn, M. Roth, and R. Lawler, "Embedding Pure Data with libpd," in *Proceedings of the Fourth International Pure Data Convention*, 2011, pp. 291–301.
- [15] J. Atherton and G. Wang, "Chunity: Integrated Audio-visual Programming in Unity," in *Proceedings of the 2018 Conference on New Interfaces for Musical Expression*, 2018.
- [16] Microsoft. (2019) Windows Mixed Reality. [Online]. Available: <https://www.microsoft.com/en-ca/windows/windows-mixed-reality>
- [17] SteamVR. (2019) SteamVR. [Online]. Available: <https://developer.valvesoftware.com/wiki/SteamVR>
- [18] VRTK. (2019) VRTK - Virtual Reality Toolkit. [Online]. Available: <https://vrtoolkit.readme.io/>
- [19] T. Machover and J. T. Chung, "Hyperinstruments: Musically intelligent and interactive performance and creativity systems," in *Proceedings of the 1989 International Computer Music Conference*, 1989.
- [20] D. Overholt, "The overtone violin," in *Proceedings of the 2005 Conference on New Interfaces for Musical Expression*, 2005, pp. 34–37.
- [21] S. Trail, M. Dean, G. Odowichuck, T. F. Tavares, P. F. Driessen, W. A. Schloss, and G. Tzanetakis, "Non-invasive sensing and gesture control for pitched percussion hyper-instruments using the kinect," in *Proceedings of the 2012 Conference on New Interfaces for Musical Expression*, 2012.
- [22] D. Johnson, I. Dufour, G. Tzanetakis, and D. Damien, "Detecting pianist hand posture mistakes for virtual piano tutoring," in *Proceedings of the 2016 International Computer Music Conference*, 2016.
- [23] T. Dwyer, K. Marriott, T. Isenberg, K. Klein, N. Riche, F. Schreiber, W. Stuerzlinger, and B. H. Thomas, *Immersive Analytics: An Introduction*. Cham: Springer International Publishing, 2018, pp. 1–23.
- [24] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, July 2002.
- [25] L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, 2008.
- [26] R. Sicat, J. Li, J. Choi, M. Cordeil, W. Jeong, B. Bach, and H. Pfister, "Dxr: A toolkit for building immersive data visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 715–725, Jan 2019.
- [27] R. Fiebrink, D. Trueman, and P. R. Cook, "A meta-instrument for interactive, on-the-fly machine learning," in *Proceedings of the 2009 Conference on New Interfaces for Musical Expression*, 2009, pp. 280–285.



# NO STRINGS ATTACHED: FORCE AND VIBROTACTILE FEEDBACK IN A GUITAR SIMULATION

**Andrea Passalenti**

University of Udine

passalenti.andrea@spes.uniud.it

**Razvan Paisa**

Aalborg University

rpa@create.aau.dk

**Niels C. Nilsson**

Aalborg University

ncn@create.aau.dk

**Nikolaj S. Andersson**

Aalborg University

nsa@create.aau.dk

**Federico Fontana**

University of Udine

federico.fontana@uniud.it

**Rolf Nordahl**

Aalborg University

rn@create.aau.dk

**Stefania Serafin**

Aalborg University

sts@create.aau.dk

## ABSTRACT

In this paper we propose a multisensory simulation of plucking guitar strings in virtual reality. The auditory feedback is generated by a physics-based simulation of guitar strings, and haptic feedback is provided by a combination of high fidelity vibrotactile actuators and a Phantom Omni haptic device. Moreover, we present a user study ( $n=29$ ) exploring the perceived realism of the simulation and the relative importance of force and vibrotactile feedback for creating a realistic experience of plucking virtual strings. The study compares four conditions: no haptic feedback, vibrotactile feedback, force feedback, and a combination of force and vibrotactile feedback. The results indicate that the combination of vibrotactile and force feedback elicits the most realistic experience, and during this condition, the participants were less likely to inadvertently hit strings after the intended string had been plucked. Notably, no statistically significant differences were found between the conditions involving either vibrotactile or force feedback, which points towards an indication that haptic feedback is important but does not need to be high fidelity in order to enhance the quality of the experience.

## 1. INTRODUCTION

In recent years, the availability of relatively low cost virtual reality (VR) hardware devices has seen applications also in the music industry. Several VR musical instruments have been developed both in the academic and commercial world. An overview of design guidelines and applications of VR musical instruments can be found in [1].

In the computer music community, the sounds of stringed instrument have been simulated for decades, starting with the work of Hiller and Ruiz [2], followed a decade later

by the simulations proposed by Karplus and Strong [3]. Moreover, physics-based simulation of such instruments has been an active areas of research within the community (see e.g. [4, 5]). However, this research has predominantly focused on simulating the sounds generated during interaction with strings, and visual and haptic feedback remain relatively unexplored (for a recent exception see [6]).

In this paper we propose a novel multisensory simulation of a guitar, which uses efficient yet accurate physics-based synthesis techniques to reproduce the auditory and haptic feedback accompanying the act of plucking guitar strings. In the current paper, we use the term *haptic* in a broad sense to encompass all somatosensory capabilities; that is, sensations that qualify as cutaneous (related to interactions at the level of the skin), kinesthetic (related to movements of one's body and limbs), and proprioceptive (related to the position of limbs and the static attitude of the musculature) [7, 8].

We first describe the system, which simulates the sensation of plucking guitar strings through a combination of visual, auditory, force and vibrotactile feedback. Subsequently, we present a user study evaluating users' experience of plucking virtual strings. The aim of the study was twofold: (1) It was meant to determine the degree of perceived realism of the system; that is the degree to which the system was able to replicate the sensation of plucking a real guitar string. (2) The aim was to explore the relative importance of force and vibrotactile feedback as elements for the creation of a realistic experience of plucking virtual strings. Specifically, it was considered relevant to determine if a realistic experience can be elicited when the simulation is devoid of force feedback and only involves vibrotactile feedback.

## 2. RELATED WORK

In the last two decades, haptic feedback has received increasing interest from the sound and music computing community, due to the strong correlation between auditory and haptic musical signals. In fact, both signals share highly similar and simultaneous temporal analogies, with a higher

Copyright: © 2019 Andrea Passalenti et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

sampling rate for the audio channel. For a recent overview refer to the work of Papetti and Saitis [9]. Avanzini and colleagues [6,10] describe a multimodal architecture, which integrates physically-based audio and haptic models with visual rendering. Experiments with stiffness perception showed how auditory feedback can modulate tactile perception of stiffness. In her PhD dissertation investigating the role of haptic feedback in digital musical instruments, O'Modhrain [11] pioneered research on multisensory audio-haptic simulations in a musical context. As an example, she discovered that the playability of touch-free instruments, such as the Theremin, is significantly increased when haptic feedback is provided. Custom made devices to provide haptic feedback in a musical context have been developed together with physics-based audio-visual simulations in the work of Florens and colleagues [12]. More recently, Leonard and Cadoz [13] introduced a system, based on mass-interaction physical modelling, that supports real-time interaction with multisensory virtual music instruments. The combination of physically simulated strings and haptic feedback has also been rather extensively explored by Berdahl [14]. In this context, the applications have been mostly pedagogical and artistic, rather than targeted towards perceptual evaluations of auditory-haptic interactions. Berdahl proposes novel fader-based controllers where the plucking action can be felt while interacting with a virtual string [15]. The tight multisensory coupling between hearing and touch has also been explored in the simulations proposed by Liu and Ando [16].

In the context of VR, although research has primarily focused on providing realistic auditory and visual feedback, haptic feedback has also been investigated. However, the focus has to a large extent been on cumbersome and expensive force feedback devices [17].

Kinesthetic and proprioceptive information is central to perception of solid objects, but so is cutaneous information derived from vibrations. The vibrations generated when an object moves across a surface encodes roughness [18] and the vibrations generated during tapping encodes hard-

ness [19]. For these reasons, much research has focused on increasing the realism of virtual interaction using vibrations [20]. For example, it has been shown that the addition of vibrotactile feedback can enhance low-fidelity kinesthetic devices [19], vibrotactile feedback affect perceived hardness during tapping of one physical object on another [21], and vibrotactile feedback can be used to elicit an illusion of compliance when pressing a stylus against a rigid surface [22].

### 3. MULTISENSORY SIMULATION OF PLUCKING GUITAR STRINGS

The system proposed in the current paper is created with the intention of eliciting a realistic sensation of plucking the strings of a virtual guitar. The system consists of three separate stimuli elements: haptic, auditory and visual. Figure 2 shows a diagram visualizing how the elements have been connected, and Figure 1 shows a user interacting with the system.

Specifically, the haptic feedback is provided by a Sensable Phantom Omni haptic device mounting a 3D printed plectrum on its arm tip. The plectrum is embedded with a Haptuator Mark II vibrotactile actuator manufactured by Tactile Labs. Visual feedback is delivered using an Oculus Rift CV1 head mounted display. Finally, auditory feedback is provided through a Vox HC30 guitar amplifier.

The signal driving the vibrotactile actuator is produced by an impact model run by the Sound Design Toolkit for Max/MSP [23]. Specifically, the contact force depends on the velocity and displacement of the objects in contact according to the following relationship:

$$f(x(t), v(t)) = kx(t)^\alpha + \lambda x(t)^\alpha v(t)$$

for  $x \geq 0$ , and 0 otherwise, where the compression  $x$  at the contact point is defined as the differences between the



Figure 1. A user interacting with the system. On the top right a zoomed figure of the Phantom Omni device with attached the vibrotactile actuator. The actuator is inserted into a 3D printed plectrum.

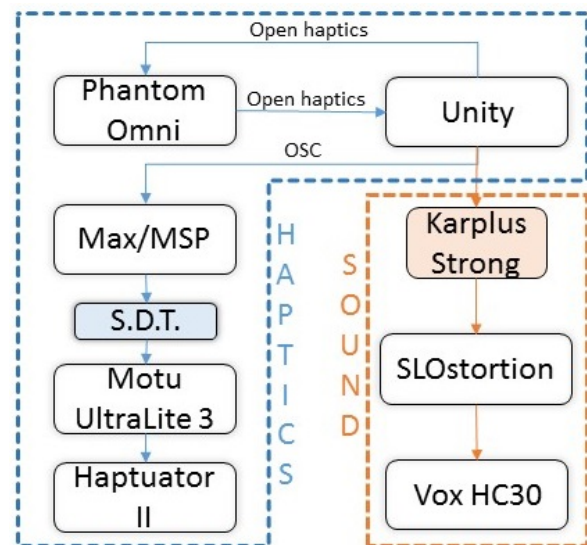


Figure 2. Diagram visualizing the software and hardware integration.

displacements of the two bodies, and  $v(t)$  is the compression velocity. The parameter  $k$  is the force stiffness and is a function of the mechanical properties of the two bodies, while  $\lambda$  is the force damping weight, and  $\alpha$  is a parameter whose value depends on the geometry of the contact [6].

The parameters of the impact model were chosen by having two guitar players empirically experiment with different settings and choosing the parameters that felt closest to a guitar pluck. For the force feedback, the Phantom Omni haptic device is driven by a collision algorithm developed in the haptic plugin for Unity [24]. This plugin uses the Open Haptics toolkit [25] to render the contact force between the pen of the Phantom Omni and the virtual string.

The guitar sound is synthesized by an efficient extended Karplus-Stong algorithm [26], originally developed by Kevin Karplus and Alex Strong in 1983 [3]. The algorithm simulates a string in the form of a feedback digital delay whose length represents the length of the string. Propagation losses are simulated using a low-pass filter. Jaffe and Smith [26] proposed improvements towards a realistic guitar sound. Although this simulation is a simplification compared to accurate physical simulations, it is efficient enough to run in real time with minimum CPU load. This fact is especially important in VR applications. In order to fully simulate the sound of an electric guitar the output of the Karplus-Stong algorithm was passed through a Wampler SLOstortion high gain drive pedal before being played back through a VOX HC-30 guitar combo amplifier. The pedal was set on overdrive mode with parameters matching the sound of the real guitar present in the training session, that was connected to the same amplifier, but on a different channel.

The visual stimuli presented an electric guitar which was created using the Unity 3D and displayed using an Oculus Rift CV1 head mounted display.

## 4. METHOD AND MATERIALS

As suggested, the aim of the study was to (1) evaluate the perceived realism of the proposed system and (2) to explore the relative importance of force and vibrotactile feedback as elements for the creation of a realistic experience of plucking strings while interacting with a virtual guitar. To meet this aim, we performed a within-subjects study comparing four conditions that varied in terms of the haptic feedback provided when users plucked virtual strings: no haptic feedback (N), vibrotactile feedback (V), force feedback (F), and a combination of force and vibrotactile feedback (FV).

### 4.1 Participants

A total of 29 participants (26 male, 3 female) aged between 19-44 years ( $M=28.2$  years,  $SD=7.0$ ) took part to the study. All participants were faculty or students at Aalborg University Copenhagen. On average, the participants had 8.2 years ( $SD=8.3$ ) of regular, weekly practice playing a music instrument, they played 2.4 hours ( $SD=2.6$ ) each week, and 21 participants reported being able to play one or more

string instruments. All participants gave written informed consent prior to participation.

### 4.2 Procedure and Task

Initially the participants completed a questionnaire covering demographic information (i.e., age, gender, occupation, and musical experience). They were then introduced to the setup and task. They were informed that the study was exploring the perceived realism of virtual strings and were instructed to pay particular attention to the haptic sensations experienced during each condition. No information was provided about the variations in feedback across conditions).

Because the aim of the study was to explore changes in realism across the four conditions, the participants were asked to pluck the strings of a real guitar before exposure to the first condition. They were instructed to pluck all six strings and were allowed to do so for no more than three minutes. It was made explicit to the participants that this task was meant as a baseline for comparison during the four conditions, and that they should pay attention to the sensation of touching the real strings, including sense of stiffness (i.e., the strings resistance to deformation).

During each condition, the participants were required to pluck each of the six strings twice in randomized order. The string the participants should pluck was visually highlighted. Subsequently, the participants were asked to freely interact with the virtual strings and they were encouraged to both pluck individual strings and perform strumming interactions. After exposure to each condition the participants were required to fill out a questionnaire related to their experience (see Section 4.3).

The participants were exposed to the four conditions in randomized order, and the study lasted for approximately 20 minutes in total.

### 4.3 Measures

Because the primary aim of the study was to determine how realistic the participants found the four conditions, we primarily relied on self-reported measures. Specifically, after exposure to each condition the participants were asked to fill out a questionnaire including eight items related to their experience of interacting with the virtual strings. The eight items can broadly be divided into four categories:

- *Perceptual similarity*: Two items required the participants to explicitly compare the real and virtual strings in terms of (a) overall similarity and (b) stiffness.
- *Perceived realism*: Three items asked the participants to evaluate (c) the overall experience of realism, (d) the sensation of touching physical strings, and (e) the sensation of hearing physical strings.
- *Perceived thickness*: Two items asked the participants evaluate (f) the connection between the thickness of the virtual strings and the sensation of touching them, and (g) the connection between the thickness of the virtual strings and sounds they generated.

Table 1. The eight questionnaire items and corresponding anchors of the 7-point (1-7) rating scales.

Questionnaire items:	Scale anchors:
<i>Perceptual similarity:</i>	
(a) The sensation of touching the virtual and real strings was:	Completely different / Identical
(b) Compared to the real strings, the stiffness of the virtual strings was:	Much lower / Much higher
<i>Perceived realism:</i>	
(c) I found the experience of interacting with the virtual guitar realistic.	Strongly disagree / Strongly agree
(d) It felt as if I was touching physical strings.	Strongly disagree / Strongly agree
(e) I felt like I was hearing physical strings.	Strongly disagree / Strongly agree
<i>Perceived thickness:</i>	
(f) It felt as if there was a connection between the thickness of the strings and how they felt.	Strongly disagree / Strongly agree
(g) It felt as if there was a connection between the thickness of the strings and the sounds.	Strongly disagree / Strongly agree
<i>Perceived ease of use:</i>	
(h) I found it easy to pluck the strings of the virtual guitar.	Strongly disagree / Strongly agree

- *Perceived ease of use:* Finally, one item (h) asked the participants to evaluate how easy they found it to interact with the virtual guitar.

All eight questions were answer using 7-point rating scales, ranging from 1 to 7. Table 1 presents the eight questions and the corresponding scale anchors.

In addition to the questionnaire administered after each four conditions, we also asked the participants to indicate which of the four the preferred once they had tried them all. Moreover, the participants were encouraged to explain their preference.

Finally, to determine whether the addition of more haptic feedback would positively affect the participants ability to pluck the virtual strings, we logged the number of erroneously plucked strings during the part of each trial where the participants had to pluck predefined strings. Impacts between the virtual plectrum and strings were considered errors, if they occurred after the correct string had been plucked and before a new string was highlighted. We deliberately ignored errors made prior to the participants plucking the correct string, as these errors were more likely to result from visual misperception. That is, errors occurring prior to initial contact with a highlighted string were likely the result of incorrect visuomotor coordination, rather than the sensation of the haptic stimuli itself. Conversely, errors occurring after a highlighted string had been plucked could be the result of an inability to perceive the haptic stimuli produced while plucking.

## 5. RESULTS

This section presents the results obtained from the self-reported measures pertaining to the participants' experience and the behavioral measure related to the number of errors performed during exposure to each condition.

### 5.1 Self-reported measures

The data obtained from the eight questionnaire items were treated as ordinal and analyzed using Friedman tests. When statistically significant differences were found pairwise comparisons using Dunn-Bonferroni tests were performed.

**Perceptual similarity:** A statistically significant difference was found in relation to *overall perceptual similar-*

*ity* ( $X^2(3) = 15.152, p = .002$ ) and the pairwise comparisons identified a statistically significant difference between N and F ( $p = .031$ ), and between N and FV ( $p = .004$ ). In both cases N yielded significantly lower scores (Figure 3a).

Similarly, a statistically significant difference was identified in regard to *stiffness relative to physical strings* ( $X^2(3) = 17.831, p < .001$ ), and the pairwise comparisons found between N and F ( $p = .003$ ), and between N and FV ( $p = .007$ ). N yielded significantly lower scores (Figure 3b). Note that both F and FV had a median score of 4, suggesting that the two conditions may have provided the greatest resemblance with the physical string in terms of stiffness.

**Perceived realism:** A statistically significant difference was found between the scores related to *overall realism* ( $X^2(3) = 11.757, p = .008$ ), and the pairwise comparisons indicated that the participants scored FV significantly higher than N ( $p = .026$ ), as apparent from Figure 3c.

The statistical comparison also indicated that the scores differed significantly with respect to the participants' *sensation of touching physical strings* ( $X^2(3) = 18.253, p = .005$ ), and the pairwise comparisons indicated significant differences between N and F ( $p = .008$ ), and between N and FV ( $p = .003$ ). Again, N yielded significantly lower scores than F and FV (Figure 3d). As indicated by Figure 3e, no significant difference was found in relation to the item pertaining to the participants' *sensation of hearing physical strings* ( $X^2(3) = 1.159, p = .763$ ).

**Perceived thickness:** A statistically significant difference was found between the scores related to the *perceived connection between string thickness and touch* ( $X^2(3) = 12.641, p = .005$ ), and the pairwise comparisons suggest that participants rated FV significantly higher than N ( $p = .019$ ), as apparent from Figure 3f. No significant difference was found with respect to the item related to *perceived connection between the thickness and the produced sound* ( $X^2(3) = 5.260, p = .154$ ).

**Perceived ease of use:** No significant difference was found between the scores related to *perceived ease of use* ( $X^2(3) = 2.026, p = .567$ ), which are summarized in Figure 3h.

**Preference rating:** When asked to select their preferred condition technique 41.4% (12 participants) chose FV, 37.9%



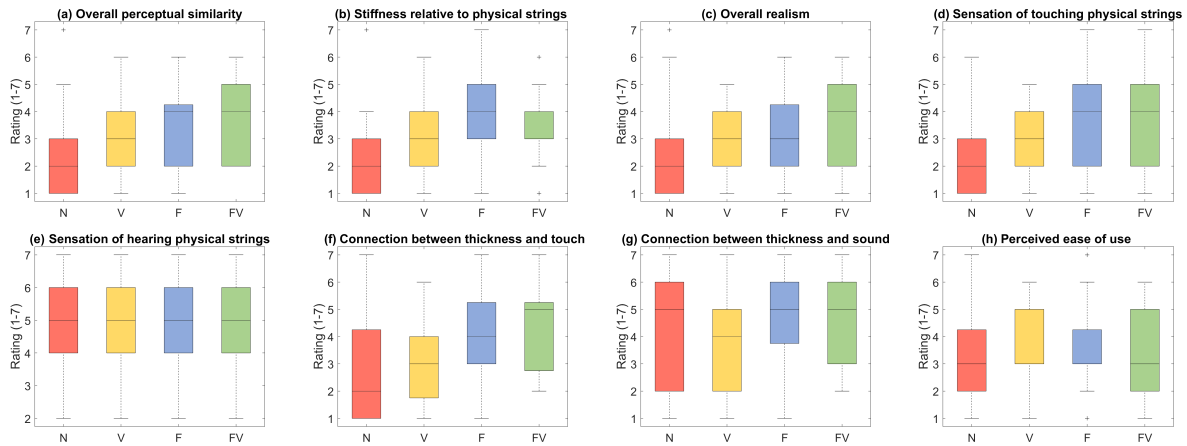


Figure 3. Boxplots visualizing the results related to the eight questionnaire items in terms of medians, interquartile ranges, minimum and maximum ratings, and outliers.

(11 participants) chose F, 10.3% (3 participants) chose V, 3.4% (1 participants) chose N, and 6.9% (2 participants) had no preference. A Cochran's Q test was run to determine if the percentages of participants who chose each condition differed. Sample size was adequate to use the  $\chi^2$ -distribution approximation. The Cochran's Q test suggested that the difference was statistically significant ( $\chi^2(4) = 19.103, p = .001$ ). Pairwise comparisons using Dunn-Bonferroni tests revealed significant differences between FV and N ( $p = .012$ ), and between F and N ( $p = .033$ ).

## 5.2 Number of Errors

One participant was excluded from this analysis because the data obtained during one of the four conditions was corrupted. The results of the behavioural measure of the number of erroneously plucked strings was treated as interval data. However, the data did not meet the assumption of normality, as assessed by Shapiro-Wilk test ( $p > .05$ ), and significant outliers were identified, as apparent from the boxplot in Figure 4. Thus, the data was analyzed using non-parametric methods. A Friedman test indicated that number of errors differed significantly between conditions ( $\chi^2(3) = 11.170, p = .011$ ) and pairwise comparisons using Dunn-Bonferroni tests indicated that FV yielded significantly fewer errors than N ( $p = .047$ ).

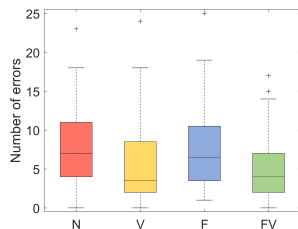


Figure 4. Boxplots visualizing the results related number of errors items in terms of medians, interquartile ranges, minimum and maximum ratings, and outliers.

## 6. DISCUSSION

The results related to *overall perceptual similarity* suggest that the sensation of plucking the virtual strings resembled its real world counterpart the most, when the simulation involved force feedback (i.e., both F and FV were significantly different from N). Based on the distribution of ratings (Figure 3a) it is apparent that some of the participants rated FV higher than F. However, no statistically significant difference between the two conditions was found. Moreover, the distributions of scores were relatively similar for F and V. It should be stressed that the median score for FV only was 4, suggesting that the participants did not experience a high degree of perceptual similarity. Future studies are necessary to determine if more elaborate string synthesis models yield more convincing results or if the scores can be attributed to limitations of the haptic rendering.

The conditions involving force feedback also provided the best match to the real guitar strings in terms of *stiffness relative to physical strings* (Figure 3b). That is, both F and FV had a median scores of 4, which indicates that the stiffness of the strings was not perceived as much higher or much lower than the stiffness of the real guitar strings.

The scores pertaining to *overall realism* (Figure 3c) indicate that the participants found the experience to be the most realistic when exposed to FV (FV was the only condition that differed significantly from N). Moreover, when the participants were asked about the degree to which they had a *sensation of touching physical strings* (Figure 3d), both F and FV yielded the highest median scores (both differed significantly from N).

The four conditions yielded largely identical and relatively high scores in relation to the self-reported *sensation of hearing physical strings* (Figure 3e). We take this to mean that most participants felt that the auditory feedback sounded as if it was generated by a physical string rather than an algorithm. It is hardly surprising that no difference was found between the four conditions, because the same auditory feedback was used across all conditions.

Based on the questionnaire item related to the perceived *connection between the thickness and touch* (Figure 3f), it would appear that the participants may have perceived the virtual strings as having different thicknesses when exposed to FV. Even though a significant difference only was found between FV and N, it is worth noting that FV is the only condition that yielded a median score higher than 4.

The results related to the *connection between thickness and sound* (Figure 3g), indicate that the participants to some extent experienced a connection between the thickness of the strings and the sound that was produced when they were plucked. However, no significant differences between conditions were found. Moreover, the spread of the scores was relatively large with respect to N and V. It is possible to offer at least two possible explanations for the large spread. That is, it is possible that the phrasing of the question prompted some participants to compare the sound to the visual appearance of the strings, while other may have compared the sound the haptic sensation of the strings. For that reason, we are reluctant to draw any conclusions from these results.

The finding that the simulations involving force feedback provided the most compelling experience is corroborated by the preference ratings and the associated qualitative feedback. That is, the majority (23/29) of the participants preferred the two conditions involving force feedback, but an almost equal number of participants preferred FV (12/29) and F (11/29). Notably, 4 out of the 11 participants who preferred F, explicitly stated that they chose F over FV because the vibration had been too strong. Of the 11 participants who preferred FV, 7 participants remarked that the vibration either made the haptic sensation of plucking the strings more realistic or added to the sense that the friction varied. Thus, the participants were somewhat conflicted about the contribution of the vibrotactile feedback, suggesting the need for future studies exploring variations in vibration intensity.

No differences were found in relation to *perceived ease of use* (Figure 3h). However, we did observe a significant difference with respect to the number of erroneously plugged strings after the correct string had been plucked (Figure 4); namely the participants plucked significantly fewer errors during FV compared to N. Moreover, even though no significant differences were found between V and the other conditions, it is worth noting that V appears to have yielded fewer errors than both N and F. In other words, the two conditions devoid of vibrotactile feedback resulted in the highest number of errors. It is possible that the added vibrations made the impact between the virtual plectrum and string more salient, and thus causing the participants to retract their hands more swiftly.

## 7. CONCLUSION

In this paper we proposed a system that allows users to pluck virtual guitar strings while receiving multisensory feedback in response to this interaction. The system was evaluated in a user study exploring the perceived realism of the simulation and the relative importance of force and vibrotactile feedback. The results indicate that the two con-

ditions involving force feedback provide the highest degree of perceptual similarity to real guitar strings. While no significant differences were found between the two conditions, the condition including both force and vibrotactile feedback yielded scores indicating that it was the best at mimicking interaction with real guitar strings. The self-reported measures related to overall realism yielded similar indications. However, the participants scored the two approaches similarly when they were asked to what extent they felt that they touched real strings. The absence of significant differences between the three haptic conditions, makes it uncertain whether force or vibrotactile is the most important for a realistic experience. Nevertheless, judging by the distribution of scores and the preference ratings, force feedback appears to be central to the participants' experience of realism, and we suspect that the combination of force and vibrotactile feedback may serve as the best proxy for physical strings. It is encouraging that the vibrotactile condition generally scored higher than the condition devoid of any haptic feedback. However, future studies involving a wider range of vibrotactile feedback are necessary in order to determine if vibrotactile feedback will suffice in and of itself. Particularly, it is necessary to compare variations in the algorithm rather than just comparing the presence and absence of vibrotactile feedback. Finally, no differences were observed with respect to perceived ease of use, but the behavioral measure provides some indication that vibrotactile feedback may decrease the risk of accidentally hitting strings after the intended string has been plucked. Thus, even if vibrotactile feedback may be less important than force feedback with respect to perceived realism, it is possible that it can help reduce the number of incorrectly plucked strings.

## 8. REFERENCES

- [1] S. Serafin, C. Erkut, J. Kojs, N. C. Nilsson, and R. Nordahl, "Virtual reality musical instruments: State of the art, design principles, and future directions," *Computer Music Journal*, vol. 40, no. 3, pp. 22–40, 2016.
- [2] L. Hiller and P. Ruiz, "Synthesizing musical sounds by solving the wave equation for vibrating objects: Part 2," *Journal of the Audio Engineering Society*, vol. 19, no. 7, pp. 542–551, 1971.
- [3] K. Karplus and A. Strong, "Digital synthesis of plucked-string and drum timbres," *Computer Music Journal*, vol. 7, no. 2, pp. 43–55, 1983.
- [4] C. Erkut, V. Välimäki, M. Karjalainen, and M. Laurson, "Extraction of physical and expressive parameters for model-based sound synthesis of the classical guitar," in *Audio Engineering Society Convention 108*. Audio Engineering Society, 2000.
- [5] G. Derveaux, A. Chaigne, P. Joly, and E. Bécache, "Time-domain simulation of a guitar: Model and method," *The Journal of the Acoustical Society of America*, vol. 114, no. 6, pp. 3368–3383, 2003.

- [6] F. Avanzini and P. Crosato, "Haptic-auditory rendering and perception of contact stiffness," in *International Workshop on Haptic and Audio Interaction Design*. Springer, 2006, pp. 24–35.
- [7] G. Robles-De-La-Torre, "The importance of the sense of touch in virtual and real environments," *Ieee Multimedia*, vol. 13, no. 3, pp. 24–30, 2006.
- [8] D. Waller and E. Hodgson, "Sensory contributions to spatial knowledge of real and virtual environments," in *Human Walking in Virtual Environments*, F. Steinicke, Y. Visell, J. Campos, and A. Lécuyer, Eds. Springer, 2013, pp. 3–26.
- [9] S. Papetti and C. Saitis, "Musical haptics: Introduction," in *Musical Haptics*. Springer, 2018, pp. 1–7.
- [10] F. Avanzini and P. Crosato, "Integrating physically based sound models in a multimodal rendering architecture," *Computer Animation and Virtual Worlds*, vol. 17, no. 3-4, pp. 411–419, 2006.
- [11] S. OModhrain and C. Chafe, "Incorporating haptic feedback into interfaces for music applications," in *Proceedings of the International Symposium on Robotics with Applications, World Automation Conference*, 2000.
- [12] J.-L. Florens, "Expressive bowing on a virtual string instrument," in *International Gesture Workshop*. Springer, 2003, pp. 487–496.
- [13] J. Leonard and C. Cadoz, "Physical modelling concepts for a collection of multisensory virtual musical instruments," in *New Interfaces for Musical Expression 2015*, 2015, pp. 150–155.
- [14] E. Berdahl and J. O. Smith III, "A tangible virtual vibrating string," in *Proceedings of the 2008 Conference on New Interfaces for Musical Expression (NIME08)*, 2008.
- [15] E. J. Berdahl, *Applications of feedback control to musical instrument design*. Stanford University, 2010.
- [16] J. Liu and H. Ando, "Hearing how you touch: Real-time synthesis of contact sounds for multisensory interaction," in *Human System Interactions, 2008 Conference on*. IEEE, 2008, pp. 275–280.
- [17] G. C. Burdea, "Force and touch feedback for virtual reality," 1996.
- [18] S. J. Lederman, R. L. Klatzky, C. L. Hamilton, and G. I. Ramsay, "Perceiving surface roughness via a rigid probe: Effects of exploration speed and mode of touch," 1999.
- [19] K. J. Kuchenbecker, J. Fiene, and G. Niemeyer, "Improving contact realism through event-based haptic feedback," *IEEE transactions on visualization and computer graphics*, vol. 12, no. 2, pp. 219–230, 2006.
- [20] H. Culbertson, S. B. Schorr, and A. M. Okamura, "Haptics: The present and future of artificial touch sensation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 385–409, 2018.
- [21] T. Hachisu, M. Sato, S. Fukushima, and H. Kajimoto, "Augmentation of material property by modulating vibration resulting from tapping," in *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*. Springer, 2012, pp. 173–180.
- [22] J. Kildal, "3d-press: haptic illusion of compliance when pressing on a rigid surface," in *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*. ACM, 2010, p. 21.
- [23] S. D. Monache, P. Polotti, and D. Rocchesso, "A toolkit for explorations in sonic interaction design," in *Proceedings of the 5th audio mostly conference: a conference on interaction with sound*. ACM, 2010, p. 1.
- [24] M. Poyade, M. Kargas, and V. Portela, "Haptic plugin for unity," *Digital Design Studio (DDS), Glasgow School of Art, Glasgow, United Kingdom*. <https://core.ac.uk/download/pdf/28875009.pdf>, 2014.
- [25] B. Itkowitz, J. Handley, and W. Zhu, "Theopenhaptics/spl trade/toolkit: a library for adding 3d touch/spl trade/navigation and haptics to graphics applications," in *Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2005. World Haptics 2005. First Joint*. IEEE, 2005, pp. 590–591.
- [26] D. A. Jaffe and J. O. Smith, "Extensions of the karplus-strong plucked-string algorithm," *Computer Music Journal*, vol. 7, no. 2, pp. 56–69, 1983.

# RaveForce: A Deep Reinforcement Learning Environment for Music Generation

**Qichao Lan**  
RITMO

Department of Musicology  
University of Oslo  
qichao.lan@imv.uio.no

**Jim Tørresen**  
RITMO

Department of Informatics  
University of Oslo  
jimtoer@ifi.uio.no

**Alexander Refsum Jensenius**  
RITMO

Department of Musicology  
University of Oslo  
a.r.jensenius@imv.uio.no

## ABSTRACT

RaveForce is a programming framework designed for a computational music generation method that involves audio sample level evaluation in symbolic music representation generation. It comprises a Python module and a SuperCollider quark. When connected with deep learning frameworks in Python, RaveForce can send the symbolic music representation generated by the neural network as Open Sound Control messages to the SuperCollider for non-real-time synthesis. SuperCollider can convert the symbolic representation into an audio file which will be sent back to the Python as the input of the neural network. With this iterative training, the neural network can be improved with deep reinforcement learning algorithms, taking the quantitative evaluation of the audio file as the reward. In this paper, we find that the proposed method can be used to search new synthesis parameters for a specific timbre of an electronic music note or loop.

## 1. INTRODUCTION

In a computational music generation task, what is essentially generated? This question leads to a debate on either to generate music in symbolic music representation, e.g. MIDI (Music Instrument Digital Interface) or to generate the audio waveform directly. Symbolic music representations can generally reflect the idiosyncrasy of a music piece, but they can hardly trace detailed music information, such as micro-tonal tunings, timbre nuances and micro-timing. Signal-based music representations are better at preserving micro-level details that are not captured well by the symbolic representations. Thus signal-based workflows—including raw audio generation—may be a solution for computational music generation. However, since raw audio generation requires much more computational resources than symbolic representation methods, there are still some difficulties for this method to generate long multi-track music pieces [1]. Furthermore, without a symbolic representation, these methods can be too sophisticated to explain from a music-theoretical perspective. Hence, our

motivation is to find a balance between these two forms of music representation in computational music generation.

Our research question is: how can an A.I system be trained to consider the music sound while generating symbolic music representation? Technically speaking, we hope that the neural network in an A.I system can not only generate symbolic sequences but also convert the symbolic representation into an audio waveform that can be evaluated. To do so, we need to use non-real-time synthesis for the transformation from symbolic music representation to an audio file which will become the input of the neural network, and the output will be accordingly the next symbolic representation. Compared with pure symbolic generation, this method also outputs the corresponding audio waveform, which may broaden the application fields. Besides, different from raw audio generation, we fix the transforming function for the neural network, which may make the computational resource focus more on the target music information than on the function estimation.

In this paper, we will explain the proposed method and provide a programming implementation as well as two simplified music tasks as examples. We start with the background of deep learning music generation in Section 2, demonstrating the relationship between the data type and the neural network architecture. In Section 3, we present our method to improve the symbolic representation and the reason why we choose to use deep reinforcement learning. Section 4 introduces the implementation details of our deep reinforcement learning environment with an emphasis on how we optimise it for a musical context. Section 5 describes the reward function design in customised tasks and explains the evaluation from running time and music quality perspective. In Section 6, we summarise the innovations and limitations of our method as well as our future directions.

## 2. BACKGROUND

Computational music generation has for a long time been an intriguing topic for musicologists and computer scientists [2]. Of current algorithmic methods, deep learning seems to be particularly relevant for music generation tasks [3]. Deep learning is a method that learns from data representations, so in terms of music generation, it is essential to study the background of how the music representation influences the learning process and result.

Copyright: © 2019 Qichao Lan et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



## 2.1 Symbolic vs signal-based representations

Music can typically be represented as either signals (audio) or symbols (score representations). Popular symbolic representation methods include MIDI, musicXML, MEI, and others [4]. Among them, MIDI is one of the most popular data formats being used in deep learning music generation tasks. In some particular styles of music, and particularly the ones based on traditional music notation, MIDI data can be an efficient representation. One example is the piano score generation in the DeepBach project [5]. Another example is that of machine-assisted composition applications, in which MIDI allows for editable features [6]. However, as mentioned in the introduction, there are also many cases in which symbolic representations are inadequate in capturing the richness and nuances of the music in question.

One way to address limitations of symbolic representations is the use of sample-level music generation, as demonstrated in WaveNet [7] and WaveRNN [8]. However, although some progress has been made, the raw audio generation requires a lot of computational resources, and it is too complicated to explain how these samples get organised from a musicology perspective.

The data format can also influence the design of the neural network. In symbolic representations, supervised learning can be found in many applications [9]. For raw audio signals, unsupervised learning techniques such as autoencoder and generative adversarial network (GAN) are frequently adopted [10, 11].

## 2.2 Reinforcement learning

Reinforcement learning is different from supervised or unsupervised learning techniques in that its updating strategy relies on the interaction between an agent and the environment rather than the function gradient. In a given period—that is, an *episode* in reinforcement learning—the agent will try to maximise the reward it can get. The reward is calculated in each episode, and it is used to update the parameters of the agents [12].

The connection between reinforcement learning and music generation goes back to the use of Markov models in algorithmic composition. As one of the pioneers in automated music generation, in the piece called *Analogique A*, Iannis Xenakis uses Markov models for the order of musical sections [13]. The use of Markov models in composition reveals its connection with reinforcement learning as the action of the agent only depends on the current state. However, in previous research on reinforcement learning in computational music generation [14], the reward function calculation is not based on the sample-level evaluation.

Recently, deep learning technology has brought new possibilities to reinforcement learning as it allows the agents to examine higher-level information. In deep reinforcement learning, the agent can be represented by a neural network, which makes it capable of evaluating the raw audio signal and then output the decision. Deep reinforcement learning has been a success during the past few years since it shows that a virtual agent can surpass human beings in several

tasks, e.g. Atari games [15]. After that, there appear more and more algorithms such as Proximal Policy Optimization (PPO) [16]. For testing these algorithms, there are many simulation environments, e.g. the OpenAI Gym<sup>1</sup>. For music, deep reinforcement learning has been used for the score following [17]. However, there is still no environment designed for music generation.

## 3. DESIGN CONSIDERATION

Though symbolic representations have shown some limitations, generating music at the audio sample level can be computationally expensive. Therefore, we propose to generate the symbolic representation first, and then use these representations to synthesise audio for evaluation.

### 3.1 From symbolic notation to audio

Our first step is to choose a proper method to convert a symbolic representation to an audio file. Three options are considered:

1. to send the generated sequence to an instrument and record the sound for evaluation.
2. to use other general-purpose programming languages such as C++ for the sound synthesis.
3. to use music programming languages like Max/MSP, Pure Data, Csound and SuperCollider for non-real-time synthesis.

We exclude the first option because it would be too time-consuming, considering there would be a considerable number of iterations in the deep learning training process. The second option is the most efficient in synthesis speed, but it lacks the extensibility from a music perspective as users have to be familiar with the C-style programming languages. The third option best balances the efficiency and usability as music programming languages have already been ubiquitous in the electronic music field [18].

However, both the second and the third option are faced with the same challenge—the *gradient*. In supervised learning, we need to know all the functions and their gradient. After comparing the output of the neural network and the training data, we should fine-tune the parameter of the neural network to minimise the loss with the help of these gradients [19]. In our proposed method, since we involve the non-real-time synthesis, back-propagation cannot be done in this context as the functions used for transforming symbolic representation to audio files are unknown.

### 3.2 Addressing the gradient problem with deep reinforcement learning

Deep reinforcement learning can solve the gradient problem mentioned above as it relies only on the interaction reward rather than the gradient. Though we cannot get the gradient from the symbolic-to-audio transforming function, We can quantitatively evaluate the synthesised audio to get a reward. Concretely, we train a neural network to output

<sup>1</sup> <https://gym.openai.com>

a sequence of symbolic music notations (such as the parameters for a synthesiser) and send the information to an audio programming language for non-real-time synthesis. Then, we compare the synthesised audio file with the target file, or we can use a neural network to grade the audio file directly. When an action brings a positive reward, the probability of the action should increase, and vice versa.

There are several important concepts in deep reinforcement learning that need to be defined in the music context (see Fig. 1):

- 1 *Step* refers to the process of executing what has been decided to do in the next 16<sup>th</sup> note or rest.
- 2 *Episode* refers to a series of continuous interactions before the *done* attribute turns to *true*, e.g. the end of a game. In a musical context, we use a *total-step* attribute to decide the length of an episode. Thus, it can vary from one single note to a note sequence.
- 3 *Observation-space* refers to the current state. In our musical context, we set the currently synthesised audio file to be the observation-space. In other words, the agent should be “aware” of the previous state (synthesised audio) and take the next step accordingly.
- 4 *Action-space* refers to the set of action choices for the agent. In a musical context, the action-space can be discrete (e.g. a note pitch) or continuous (e.g. the amplitude).

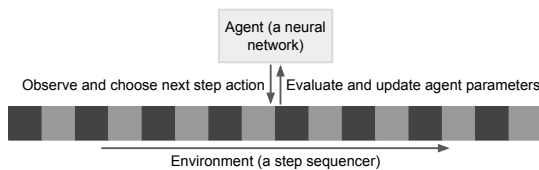


Figure 1. RaveForce architecture: in each note (step), the agent (neural network) will choose an action according to its observation on the current state (currently synthesised audio).

## 4. IMPLEMENTATION AND OPTIMISATION

As is discussed above, the key to our proposed method is to have an environment that can handle the non-real-time synthesis and evaluate the result. In our implementation of RaveForce<sup>2</sup>, we follow the OpenAI Gym interfaces in our Python module, and in SuperCollider, we create a quark to execute the non-real-time audio synthesis. In order to connect with deep learning frameworks, some optimisation is necessary for the observation space.

### 4.1 The idea from a live coding session

To implement the environment, we refer to a live coding session [20]. In many live coding sessions, SuperCollider<sup>3</sup>

has been used as the audio engine as it tracks the time and beat accurately [21]. SuperCollider employs a client-server architecture that contains two parts: the *scsynth* (SuperCollider Synthesiser) and the *sclang* (SuperCollider Language). The *sclang* will be combined in real-time to a simplified version of Open Sound Control (OSC) messages [22] and sent to the *scsynth* to control the sound. This architecture allows the *scsynth* server to run alone, while *sclang* can be replaced by other domain-specific languages (DSLs) like TidalCycles<sup>4</sup>. In short, in a live coding session, the live coders use DSLs as a client to control the real-time sound synthesis in the SuperCollider server. For our need, instead of using SuperCollider to output real-time audio signals, we use it for non-real-time audio synthesis.

As for the client, we choose to write it in Python because several deep learning frameworks (such as PyTorch<sup>5</sup>) have been implemented in Python, and the Python module Gym is one of the most important benchmarks for deep reinforcement learning. By designing the client part in Python, we can follow the Gym interface and connect with a deep learning framework, while we move the interaction part (the audio synthesis) to the SuperCollider server. With the help of Open Sound Control messages, we link the neural network training with the audio synthesis (see Fig. 2).

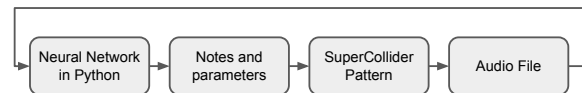


Figure 2. Python-SuperCollider communication: a neural network (agent) is trained in Python; it sends symbolic music representations (e.g. notes and synthesiser parameters) as Open Sound Control messages to the SuperCollider pattern; then the pattern will be synthesised to an audio file in non-real-time and sent back to Python as the input of the neural network, forming an iteration.

## 4.2 Code implementation

The pseudo-code of the implementation is as follows:

- 1 Use *make* function in the client to create the required environment, which will send a message to the server, asking the server to load related music patterns, synthesise an empty file and return the address of the file to the client side. On receiving the returning message, the client should read the action space and the observation space.
- 2 Send the *reset* message to the server side. Empty the observation space if it is not.
- 3 According to the observation space, decide what action to take. Send the *step* message to the server side with chosen actions in each step. The server will do non-real-time synthesis in each step according to the given

<sup>2</sup><https://github.com/chaosprint/RaveForce>

<sup>3</sup><https://supercollider.github.io>

<sup>4</sup><https://tidalcycles.org>

<sup>5</sup><https://pytorch.org>

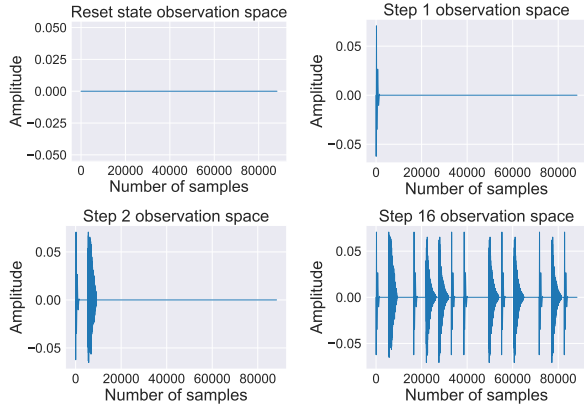


Figure 3. The observation space in each state has the same length. For instance, the step 1 only has the audio information for the first 16th note, but it is padded to have the same length as the reset state (about 90000 samples).

action message. Also, the server should return the client with the synthesised file address.

- 4 The client should use the address to load the currently synthesised sound file and set it as the observation space. Calculate the reward by comparing the generated audio file with the target audio file.
- 5 Send the reward back to the client for updating the neural network.
- 6 Repeat from Step 3 until the limit of episode length is reached

### 4.3 Optimisation

In the implementation, a unique strategy is designed for the observation space. As neural networks typically require a fixed length input, the observation space needs to be padded to have the same length in every step. Hence, in the initialisation stage, we require SuperCollider to generate an empty full-step (16-step by default) long audio file corresponding to the beats per minute (BPM) parameter. The length of this empty file will be set as the *total-length* attribute. In the following steps, though the actual output of the audio file varies in length, it will be padded with zeros to match the total-length attribute. With this strategy, the observation spaces in each step can share the same length (see Fig. 3).

## 5. TASK DESIGN AND EVALUATION

After implementing the environment, it is necessary to examine what kind of tasks it can handle and evaluate how the environment performs with the given task.

### 5.1 Challenges with the reward function design

The reward function in reinforcement learning measures how well the agent chooses the action in the current step. Its design is challenging for music generation, especially in those tasks whose evaluation criteria are subjective. It can

be feasible to evaluate the similarity between the generated music piece and the songs in a music corpus. At the same time, pursuing similarity in music can lead to plagiarism, which is an essential issue to address [23].

Currently, we provide four criteria for evaluation: (1) mean square error (MSE) of all the samples; (2) MSE of the Mel-frequency cepstral coefficients (MFCCs); (3) MSE of the short-time Fourier transform (STFT) coefficients, both real and imaginary parts; (4) MSE of the constant-Q transform (CQT) coefficients, both real and imaginary parts. These four criteria are used to measure the similarities between two audio files. Also, as the whole programming framework is customisable, it can be connected with other criteria, e.g. a well trained neural network that can grade a music file.

### 5.2 The example tasks

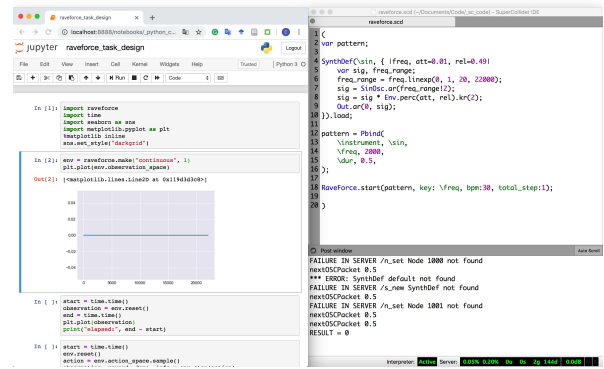


Figure 4. RaveForce workflow: first run SuperCollider code, and then open Python IDE (e.g. Jupyter Notebook) to train the agent.

In RaveForce, the music task should be defined by the user (see Fig. 4). We provide two music examples to explain the environment better.

#### 5.2.1 Drum loop imitation

The example task *drum-loop* uses music samples from three drum components (kick drum, snare drum, and hi-hat) to imitate the target drum loop as much as possible. The action space in the example is a discrete set that contains all eight possible combinations in each note from which the agent should choose one action, and a reward will be calculated according to the choice (see Fig. 5).

Different from some other reinforcement tasks, the reward in this task is precisely the state value function. If we use Deep Q-learning (DQN) for this task, the Q function in each step can be calculated as follows:

$$Q^\pi(a|s) = V(s_{t+1}) - V(s) \quad (1)$$

Also, as a specific drum combination only has a fixed reward, we can use the traditional dynamic programming method to find the best drum pattern in this case.

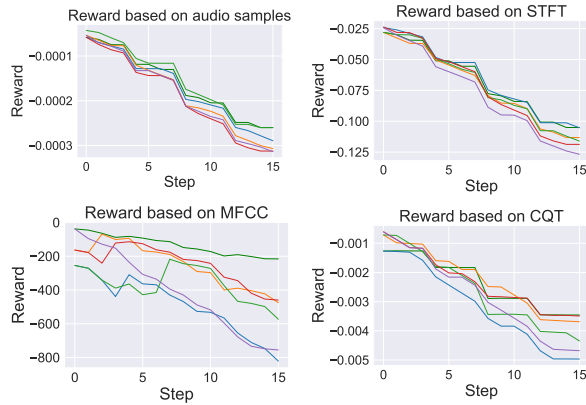


Figure 5. Drum loop combination reward with different criteria. The green line represents the reward of the optimal drum combination which is closest to the target drum loop while the rest are random combination rewards. The MFCC criterion tends to outperform others in this task.

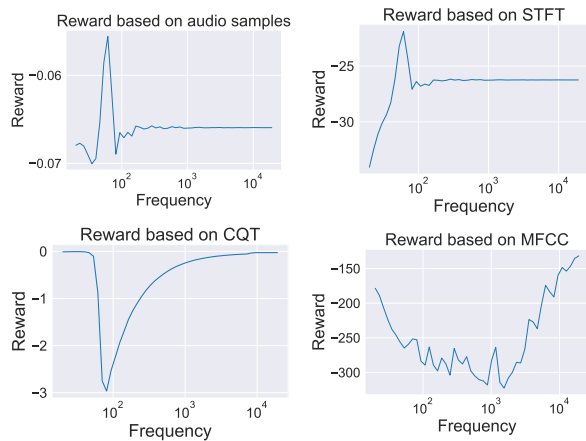


Figure 6. Different criteria for kick drum simulation task. MFCC and CQT tend to show poor performance in this task.

### 5.2.2 Kick drum optimal frequency search

In this example, we aim to use a sine wave oscillator, controlled by an amplitude envelope to simulate a kick drum audio sample. To make it easier for visualisation, we fix the envelop shape and make the frequency of the sine wave oscillator the only controllable parameter. The relationship between the frequency and the reward is shown in Fig. 6. The *total-step* attribute in SuperCollider can be set to one, which makes the pattern become a single note. In each iteration, the parameter updating of the whole loop is done for this single note. Also, the example can be extended to more parameters and more steps.

With the frequency-reward distribution, we can use the neural network to search for the optimal frequency. First, we train a critic-network which takes the frequency as input and predicts the reward. When connected with the critic-network, an actor-network can be trained until it converges to the optimal frequency.

## 5.3 Evaluation

We will evaluate the environment from two angles: (1) whether the environment is fast enough for the training; (2) if the symbolic-to-audio conversion can help the music generation.

As a support to our method, the programming framework implementation is the focal point of this paper. In previous sections, we have introduced our environment design and the optimisation we have made, which makes it feasible to use audio evaluation methods for symbolic generation within an acceptable running time. To illustrate, we provide the running time of a 16-step task in one episode (see Fig. 7), which is calculated with the *drum-loop* task mentioned above.

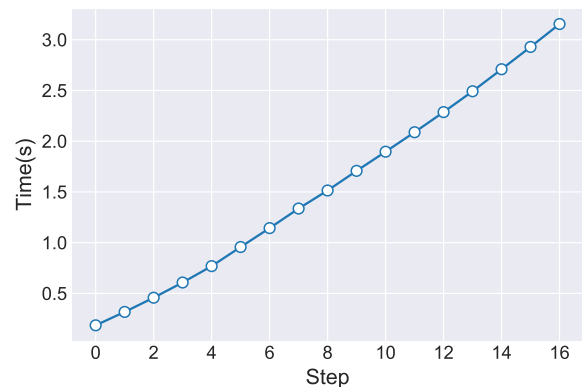


Figure 7. The running time of RaveForce example task *drum-loop*. Step 0 refers to the reset state and some time will be spent for calculating the total-length. All 16 steps will take around 3.2 seconds on an Apple MacBook Pro 13-inch (Mid 2017, i5, without Touch Bar).

Regarding the quality of the music, there are still some uncertainties, for the generated music quality may change with different algorithms, tasks and music genres. Currently, limited by computational resources, we focus mainly on the programming framework implementation, and only pay particular attention to electronic music loop or note.

Also, it is arguable that the predefinition of synthesiser architecture can be a limitation of music complexity. However, this trade-off is significant to our proposed method. With a fixed transforming function, for example, the neural network will no longer need to organise all the audio samples to form an audio waveform which is aurally similar to an FM synthesis tone. Instead, the computational resources can be used to focus on optimising the parameters of a predefined FM synthesiser. This trade-off may even bring new possibilities in music creation because mismatching the target tone with a random synthesiser architecture can potentially generate a tone which is similar but slightly different from the target.

## 6. CONCLUSION

In this project, we propose a new music generation design that employs deep reinforcement learning, and we have im-



plemented an environment for testing the design. It follows the OpenAI Gym interfaces but moves the interaction to SuperCollider. It turns out that the SuperCollider is fast enough in non-real-time audio synthesis, which makes the reward calculation and the neural network training feasible. Meanwhile, there are some uncertainties if this method can improve the music generation, which should be tested with different tasks, algorithms and music genres. It can be one of our future directions. Nevertheless, the whole implementation produces an environment for researches to explore new algorithms for music generation tasks, e.g. music sequence generation or timbre parameter searching. It provides a new perspective to music generation, especially for those tasks in which users can find a determined reward function.

### Acknowledgments

This work was partially supported by the Research Council of Norway through its Centres of Excellence scheme, project number 262762 and by NordForsk Nordic University Hub Nordic Sound and Music Computing Network NordSMC, project number 86892.

### 7. REFERENCES

- [1] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," in *Advances in Neural Information Processing Systems*, 2018, pp. 8000–8010.
- [2] D. Herremans, C.-H. Chuan, and E. Chew, "A functional taxonomy of music generation systems," *ACM Computing Surveys (CSUR)*, vol. 50, no. 5, p. 69, 2017.
- [3] J.-P. Briot, G. Hadjeres, and F. Pachet, "Deep learning techniques for music generation-a survey," *arXiv preprint arXiv:1709.01620*, 2017.
- [4] I. Fujinaga, A. Hankinson, and L. Pugin, "Automatic score extraction with optical music recognition (omr)," in *Springer Handbook of Systematic Musicology*. Springer, 2018, pp. 299–311.
- [5] G. Hadjeres, F. Pachet, and F. Nielsen, "Deepbach: a steerable model for bach chorales generation," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1362–1371.
- [6] I. Simon and S. Oore, "Performance rnn: Generating music with expressive timing and dynamics," 2017.
- [7] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR abs/1609.03499*, 2016.
- [8] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. v. d. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," *arXiv preprint arXiv:1802.08435*, 2018.
- [9] B. L. Sturm, O. Ben-Tal, U. Monaghan, N. Collins, D. Herremans, E. Chew, G. Hadjeres, E. Deruty, and F. Pachet, "Machine learning research that matters for music creation: A case study," *Journal of New Music Research*, vol. 48, no. 1, pp. 36–55, 2019.
- [10] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SAMPLRN: An unconditional end-to-end neural audio generation model," *arXiv preprint arXiv:1612.07837*, 2016.
- [11] C. Donahue, J. McAuley, and M. Puckette, "Synthesizing audio with generative adversarial networks," *arXiv preprint arXiv:1802.04208*, 2018.
- [12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [13] I. Xenakis, *Formalized music: thought and mathematics in composition*. Pendragon Press, 1992, no. 6.
- [14] N. Collins, "Reinforcement learning for live musical agents," in *ICMC*, 2008.
- [15] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.
- [16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [17] M. Dorfer, F. Henkel, and G. Widmer, "Learning to listen, read, and follow: Score following as a reinforcement learning game," *arXiv preprint arXiv:1807.06391*, 2018.
- [18] G. Wang, "A history of programming and music," *The Cambridge Companion to Electronic Music*, pp. 55–71, 2007.
- [19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.
- [20] A. McLean and R. T. Dean, "Musical algorithms as tools, languages, and partners," *The Oxford Handbook of Algorithmic Music*, p. 1, 2018.
- [21] J. McCartney, "Rethinking the computer music language: Supercollider," *Computer Music Journal*, vol. 26, no. 4, pp. 61–68, 2002.
- [22] M. Wright, "Open sound control: an enabling technology for musical networking," *Organised Sound*, vol. 10, no. 3, pp. 193–200, 2005.
- [23] A. Papadopoulos, P. Roy, and F. Pachet, "Avoiding plagiarism in markov sequence generation," in *AAAI*, 2014, pp. 2731–2737.

# Music Temperaments Evaluation Based on Triads

Tong Meihui  
JAIST  
mieto@jaist.ac.jp

Tojo Satoshi  
JAIST  
tojo@jaist.ac.jp

## ABSTRACT

It is impossible for one temperament to achieve optimally both of consonance and modulation. The dissonance level has been calculated by the ratio of two pitch frequencies, however in the current homophonic music, the level should be measured by chords, especially by triads. In this research, we propose to quantify them as Dissonance Index of Triads (DIT). We select eight well-known temperaments and calculate seven diatonic chords in 12 keys and compare the weighted average and standard deviation to quantify the consonance, and then we visualize our experimental results in a two-dimensional chart to compare the trade-offs between consonance and modulation.

## 1. INTRODUCTION

Nowadays, 12-tone equal temperament is prevalent and other temperaments has fallen to only historical and mathematical interests. However, even now in the actual performance, string and wind instruments are played in an *ad hoc* adjustment of pitches unless accompanied by keyboard instruments. In this age, those electronic instruments ease us in using any scale more freely. Then, our motivation in this paper is to give quantitative understanding to the dissonance level in various temperament in terms of triads.

The difference of temperaments has been often mentioned by the ratio of two pitch frequencies and such web site as *Pianoteq*<sup>1</sup> provides us a very convincing interface to experience the difference of temperaments; however, there were no mathematical formulation to evaluate the consonance and modulation<sup>2</sup> of a chord.

Consonance and dissonance are ambiguous psychological notions. The purpose of this research is to explore the mathematical model of Dissonance Index of Triads (DIT). In 1863, Helmholtz [1] proposed the mathematical model of consonance and dissonance in tones in terms of beats and roughness. In 1965, Plomp and Levelt [2] defined the dissonance curve between two pure tones. Later, the mathematical formula of the curve has been improved, and Vasileakis [3] claimed that the formula he proposed had been believed to be most reliable; and thus, we employ it also in this paper, though adding the effect of overtones.

<sup>1</sup> <https://www.pianoteq.com/>

<sup>2</sup> In this paper, the modulation means a key transposition.

Copyright: © 2019 Tong Meihui et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Ogata [4] has proposed the idea of consonance value among chords, calculating the consonance value of each two tones in the triads and adding them up, and then drew a 3D dissonance curve of the chords. Also, Cook [5] showed the acoustical properties of triads, claiming the perception of harmony is not simply a sum of inner consonance. In this research, we revise the Ogata's calculation and formalize the dissonance level in a more rigorous way.

This paper is organized as follows. In the following section, we show preliminaries including the introduction of various temperaments. Thereafter, we propose our formalization and show its visualization. Then, we analyze the results, and finally conclude.

## 2. PRELIMINARIES

### 2.1 Scale and Temperaments

A set of notes employed in a music piece is, when arranged in a pitch order in an octave, is called a scale. The ratio of frequency of two pitches is fixed by natural science disciplines such as physical science, acoustics, and psychology, among which mathematics plays the most important role, and one fixed series of ratios in a scale gives the notion of *temperament*.

Pythagoras in ancient Greece discovered that the perfect fifth interval with the frequency ratio of 3:2 as the most consonant, next to the octave of 2:1, around 550 BC [6]. The *Sanfen Sunyi-fa* by Jing Fang in China (BC77–BC37) is considered to have invented the same temperament with Pythagorean tuning [7].

Since then, musicologists have been constantly exploring how to solve the problem of *Pythagorean comma*, that is the error which slightly exceeds the octave when the 12th tone is introduced by multiples of 3/2. If we peremptorily regard the 12th locates at the octave, the interval between the 11th and the 12th becomes narrower than the other fifths. In later years, Pythagoras pitch was amended to place the narrow fifth, so called the wolf fifth<sup>3</sup>, between G $\sharp$  and D $\sharp$  where the fifth is rarely used.

The ultimate temperament for consonance is the just intonation, introducing the multiple by 5 in addition to 3/2, where the ratio between the intervals can be expressed all by small integers [8]. However, in contrast, the just intonation is very clumsy in modulation. The scale evolved into the mean-tone systems [9, 10], well-temperaments by Andreas Werckmeister (1645–1706) [11] or by Johann Philipp Kirnberger (1721–1783) [12], and since then there exist hundreds or even thousands of music temperaments.

<sup>3</sup> It is named after the unpleasant sound like the roaring of wolves.

	$C$	$^bD$	$D$	$^bE$	$E$	$F$	$\#F$	$G$	$^bA$	$A$	$^bB$	$B$
Sanfen Sunyi-fa	1	$\frac{3^7}{2^{11}}$	$\frac{9}{8}$	$\frac{3^9}{2^{14}}$	$\frac{81}{64}$	$\frac{3^{11}}{2^{17}}$	$\frac{729}{512}$	$\frac{3}{2}$	$\frac{2^{12}}{3^8}$	$\frac{27}{16}$	$\frac{2^{15}}{3^{10}}$	$\frac{243}{128}$
Pythagorean Tuning	1	$\frac{256}{243}$	$\frac{9}{8}$	$\frac{32}{27}$	$\frac{81}{64}$	$\frac{4}{3}$	$\frac{729}{512}$	$\frac{3}{2}$	$\frac{128}{81}$	$\frac{27}{16}$	$\frac{16}{9}$	$\frac{243}{128}$
Just Intonation	1	$\frac{16}{15}$	$\frac{9}{8}$	$\frac{6}{5}$	$\frac{5}{4}$	$\frac{4}{3}$	$\frac{45}{32}$	$\frac{3}{2}$	$\frac{8}{5}$	$\frac{5}{3}$	$\frac{16}{9}$	$\frac{15}{8}$
Quarter-Comma Meantone	1	$\frac{2^3}{5^{\frac{3}{4}}}$	$\frac{5^{\frac{1}{2}}}{2}$	$\frac{2^2}{5^{\frac{3}{4}}}$	$\frac{5}{4}$	$\frac{2}{5^{\frac{1}{4}}}$	$\frac{5^{\frac{6}{4}}}{2^3}$	$5^{\frac{1}{4}}$	$\frac{8}{5}$	$\frac{5^{\frac{3}{4}}}{2}$	$\frac{2^2}{5^{\frac{1}{2}}}$	$\frac{5^{\frac{5}{4}}}{2^2}$
Conventional QC Meantone	1	$\frac{5^{\frac{7}{4}}}{2^4}$	$\frac{5^{\frac{1}{2}}}{2}$	$\frac{2^2}{5^{\frac{3}{4}}}$	$\frac{5}{4}$	$\frac{2}{5^{\frac{1}{4}}}$	$\frac{5^{\frac{6}{4}}}{2^3}$	$5^{\frac{1}{4}}$	$\frac{5^2}{2^4}$	$\frac{5^{\frac{3}{4}}}{2}$	$\frac{2^2}{5^{\frac{1}{2}}}$	$\frac{5^{\frac{5}{4}}}{2^2}$
Werckmeister	1	$\frac{2^8}{3^5}$	$\frac{64\sqrt{2}}{81}$	$\frac{32}{27}$	$\frac{2^{13}\sqrt{2}}{3^8}$	$\frac{4}{3}$	$\frac{2^{10}}{3^6}$	$\frac{8\sqrt[4]{8}}{9}$	$\frac{128}{81}$	$\frac{2^{10}\sqrt[4]{2}}{3^6}$	$\frac{16}{9}$	$\frac{128\sqrt[4]{2}}{81}$
Kirnberger	1	$\frac{135}{128}$	$\frac{9}{8}$	$\frac{32}{27}$	$\frac{5}{4}$	$\frac{4}{3}$	$\frac{45}{32}$	$\frac{3}{2}$	$\frac{128}{81}$	$\frac{3\sqrt{5}}{4}$	$\frac{16}{9}$	$\frac{15}{8}$
Equal Temperament	1	$2^{\frac{1}{12}}$	$2^{\frac{2}{12}}$	$2^{\frac{3}{12}}$	$2^{\frac{4}{12}}$	$2^{\frac{5}{12}}$	$2^{\frac{6}{12}}$	$2^{\frac{7}{12}}$	$2^{\frac{8}{12}}$	$2^{\frac{9}{12}}$	$2^{\frac{10}{12}}$	$2^{\frac{11}{12}}$

Table 1: Ratios of Temperaments in Fractions

The equal temperament has been the product of compromise, which systematically compensated the Pythagorean comma, defining each half tone to be the 12th root of 2. Then, the temperament perfectly eased the modulation, that is, to enable us to change from one key to another freely, and was applied to the tuning of most modern musical instruments around the world. But we can never say that the equal temperament is satisfactory because it rejects the original intention of the temperament, *viz.*, the consonance between intervals. Table 1 lists the frequency ratios of some typical music temperaments introduced above.

## 2.2 Helmholtz's Theory of Beats

In physics, the superposition of two simple sinusoidal waves with similar but slightly different frequency will cause periodic fluctuation in strength through time. This phenomenon is known to piano tuners as *beats*. Hermann Helmholtz [1] concluded that dissonance is produced by the beats between two pure tones (without overtones) or between a pair of partials of two complex sounds.

When the difference in frequency is small, the beats can be easily heard. As the difference is increased to 20-30 Hz, the beats will create the impression like 'jarring and rough' described by Helmholtz. Beyond this approximate point, the beats gradually become too rapid to be identified and the sensation of roughness disappears.

## 2.3 Dissonance Curve

In 1965, Plomp and Levell confirmed Helmholtz's hypothesis by several experiments [2]. They plotted the dissonance curve and proposed the concept of critical bandwidth. Note that though the sound produced by the musical instruments has a complex timbre this psychological experiment employed only pure tones with the simplest spectrum. The combined experimental results is shown in Figure 1, and nowadays this result is widely accepted.

The figure shows the consonance/dissonance feeling when the frequency is apart from the fixed base tone. The vertical axis on the right side of the figure represents the degree

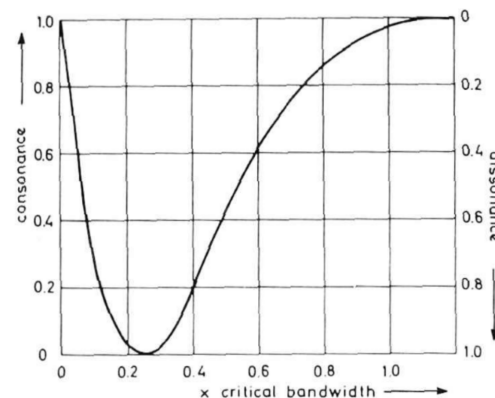


Figure 1: Dissonance Curve From a Fixed Tone to Another Tone [2]

of dissonance, and the interval is from 0 to 1 from top to bottom. The lower the vertical value is in this figure, the more dissonant. The horizontal axis is the frequency difference between higher tone and the base tone, divided by the value of the critical bandwidth. As the frequency difference gradually becomes larger, we can observe the result of the dissonance value  $d$  between the two pure tones varying. The most dissonant position ( $d = 1$ ) is said to be about a quarter of the critical band.

When the frequency of the tone is too high or too low to be heard by human ears, the identification of the tones becomes not that easy. When the horizontal axis of the dissonance curve only takes the frequency difference (without divided by critical bandwidth), we need to draw many different graphs according to the difference of base tones.

## 2.4 Numerical Calculation of Dissonance in Two Tones

Among various proposals [13–15] on the numerical calculation of dissonance, Vassilakis suggested two principal studies [16, 17], incorporating the notion of roughness [3].

Given a signal whose spectrum has two sinusoidal components with frequencies  $f_1, f_2$  and amplitudes  $v_1, v_2$ , where

$$f_{\min} = \min(f_1, f_2), f_{\max} = \max(f_1, f_2), \\ v_{\min} = \min(v_1, v_2), v_{\max} = \max(v_1, v_2),$$

the roughness (dissonance value) of  $d$  becomes:

$$d(f_1, f_2, v_1, v_2) = X^{0.1} \cdot 0.5(Y^{3.11}) \cdot Z \quad (1)$$

in which

$$X = v_{\min} \cdot v_{\max} \\ Y = 2v_{\min} / (v_{\min} + v_{\max}) \\ Z = e^{-b_1 s(f_{\max} - f_{\min})} - e^{-b_2 s(f_{\max} - f_{\min})}$$

with  $b_1 = 3.5, b_2 = 5.75$ ,

$$s = \frac{0.24}{s_1 f_{\min} + s_2}; s_1 = 0.0207; s_2 = 18.96.$$

Vassilakis has confirmed that his formula reliably and efficiently represents the perception of roughness and performs better than the preceding formulae. Therefore, the temperament evaluation model in this paper is made under this function of dissonance curve.

We generalize the roughness value (1) to include multiple, more than two sinusoidal partials as the sum of each pair of two partials. Suppose a spectrum  $F$  with fundamental frequency  $f$  is a collection of  $n$  sinusoidal waves (or partials) with frequencies  $a_1 f, a_2 f, \dots, a_n f$  and amplitudes  $v_1, v_2, \dots, v_n$ . Also, we assume that each tone contains  $n$  overtones of  $[a_1, a_2, \dots, a_n] = [1, 2, \dots, n]$ . According to [4], we also assume that  $v_1, v_2, \dots, v_n$  is a geometric progression with common ratio of 0.9; that is,  $v_1, v_2, \dots, v_n = 1, 0.9, 0.81, \dots, 0.9^{n-1}$ . So when two notes of  $F_1$  and  $F_2$  are played simultaneously, the dissonance value  $D(F_1, F_2)$  between them is

$$D(F_1, F_2) = \sum_{i=1}^n \sum_{j=1}^n d(i f_1, j f_2, v_i, v_j) \quad (2)$$

When  $F_1$  and  $F_2$  are at interval  $t$  and with the same amplitude (e.g.  $F_2 = t F_1$ ), the transposed version of  $F$  can be defined as  $tF$  with partials at  $t f, 2t f, \dots, nt f$  and amplitudes  $v_1, v_2, \dots, v_n$ . The roughness  $D_F(t)$  generated by the spectrum  $F$  is defined in function (3) and the shape of this function is shown in Figure 2.<sup>4</sup> This figure shows the comparison from a base tone to its seven overtones.

$$D_F(t) = \sum_{i=1}^n \sum_{j=1}^n d(i f, t j f, v_i, v_j), \quad (3)$$

### 3. DISSONANCE INDEX OF TRIADS

Thus far, we have introduced the preceding works concerning the dissonance value between the intervals. In this section, we propose our new definition of the dissonance value for triads. Given three tones with the ratio of intervals  $1 < t_1 < t_2$ , we add up the three values of (3) as function (4) and draw Figure 3 based on this function.

<sup>4</sup> The figure is a reproduction, appearing in [4].

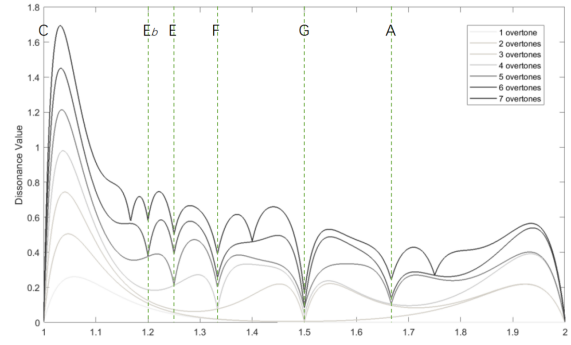


Figure 2: Dissonance Value for Intervals, Dependent on Base Frequency

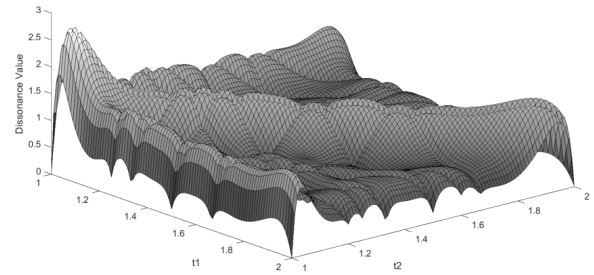


Figure 3: 3D Representation of Interval Dissonance in Triads

$$D_F(t_1, t_2) = D_F(t_1) + D_F(t_2) + D_{t_1 F}\left(\frac{t_2}{t_1}\right) \quad (4)$$

where

$$D_F(t_1) = \sum_{i=1}^n \sum_{j=1}^n d(i f, j t_1 f, v_i, v_j),$$

$$D_F(t_2) = \sum_{i=1}^n \sum_{j=1}^n d(i f, j t_2 f, v_i, v_j),$$

$$D_{t_1 F}\left(\frac{t_2}{t_1}\right) = \sum_{i=1}^n \sum_{j=1}^n d(i t_1 f, \frac{t_2}{t_1} j f, v_i, v_j).$$

We have employed twelve major keys and twelve minor keys, each of which includes seven triads on diatonic notes, including three major triads, three minor triads, and one diminished triad (vii°). In this paper, we have omitted the harmonic and melodic minor scales. Therefore, since a pair of parallel keys consists of the same set of chords, we take 12 group of chords as research objects to evaluate the music temperaments.

In the first attempt, the average value of 12 group of chords in each temperament are calculated with our dissonance value model. According to the ratios in Table 1, we consider the frequencies of an octave starting from the central C, for three typical temperaments (Pythagorean tuning, just intonation and equal temperament) as examples. The results are shown in Figure 4.



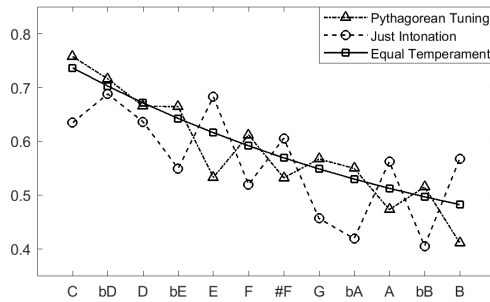


Figure 4: Relative Dissonance of Chords in a Major Scale, Compared to Equal Temperament

As we can see, even the equal temperament does not guarantee consistent values in different keys. The reason for this is obviously due to the different frequencies of the base pitches, resulting in different critical bandwidths. Therefore, in this paper, we choose to preserve the frequency ratios and to transpose the base pitch into a certain fixed value  $F_0$ . The adjusted model should no longer be called the dissonance level but an index to compare the consonance degree among different triads, and thus we call it the Dissonance Index of Triads (DIT) (5), hereafter.

$$DIT(t_1, t_2) = D_{F_0}(t_1, t_2) \quad (5)$$

We set the frequency of  $F_0$  to 263Hz, that is an approximate frequency of the central C.

#### 4. DIT IN TEMPERAMENTS

This chapter expounds the results of DIT values of triads in different temperaments. According to Figure 2, this research employs overtones upon the seven diatonic tones. We fix the ratio of amplitudes to be a geometric progression with 0.9 as mentioned before.

##### 4.1 The Weighted Chords in Each Key

Prior to the evaluation, we have given the following weights [0.86 : 0.26 : 0.17 : 0.73 : 0.73 : 0.56 : 0] on each triads on the diatonic scale, based on the usage of the chords in 1300 popular songs<sup>5</sup> after transposed to C-major. Note that our objective here is to compare the average consonant level of various chords in different keys, and not to assess the human feeling; thus the transposition is a kind of approximation. The average consonant level is shown in Table 2, for each key and temperament.

For more intuitive and clear understanding, we compare the DIT between two triads chosen from the two different temperaments. In Figure 5, the horizontal axis of each graph represents the keys while the vertical axis represents the DIT value. Since the DIT value is adjusted from the dissonance value, the lower the DIT value is the more consonant the key is.

The comparison of the Sanfen Sunyi-fa and Pythagorean tuning is shown in the upper-left graph in Figure 5. We can

see that because they share the same process of generation except for the location of the wolf fifth, the difference of DIT shifts in parallel. Similarly, the quarter-comma mean-tone and the conventional quarter-comma mean-tone, shown in the upper-right graph has the same property.

In the lower-left graph in Figure 5, we can hardly find the big difference between the two well temperaments along the horizontal trends. That is to say, the well temperaments tend to change slightly between adjacent keys, trying to distribute the dissonance reasonably to keep the balance, and thus there are no peaks in dissonance. They also especially ensures some commonly used keys such as C-major, D-major, F-major, and G-major, are in better consonance.

The lower-right graph in Figure 5 shows the comparison between the just intonation and the equal temperament. We can see that the equal temperament presents a perfect horizontal line, which proves that it will sound always the same no matter what key it is. But, its DIT value is also relatively higher with no keys in better consonance. The just intonation has the lowest DIT value of all the results in a few keys such as C and A<sup>b</sup>, but the line goes up and down steeply and the dissonant keys are also obvious. We can easily read in this figure that the equal temperament and the just intonation are the two extremes in modulation and consonance.

##### 4.2 Consonance or Modulation

In accordance with the position of the wolf fifth or other adjustments, there are also difference in what keys they prefer. In fact, there is a difference in the degree of commonality of each key, which means we had better take the weights of keys into consideration. A survey of “The Most Popular Keys of All Music”<sup>6</sup> on Spotify<sup>TM</sup> in 2005 showed the data in Table 3. Here, we put a major key and its parallel key together because they share a common set of the diatonic chords.

Taking both the weights of keys and the diatonic chords into consideration, we visualize the balance between the consonance and modulation of temperaments as in Table 4, which is plotted in Figure 6. The horizontal axis shows the average DIT value with weights, which refers to the consonance level of the temperament, while the vertical axis represents the average standard deviation of chords and represents the smoothness of modulation. The lower the value is, the more easily the temperament can modulate. It is obvious that just intonation is outstanding at consonance but worse in modulation, and equal temperament is the opposite, that is, the easiest in modulation but the worst in consonance. Pythagorean tuning and Sanfen Sunyi-fa are staying at a similar level on modulation, and there are slight difference because of the weights in keys.

At last, the well-temperaments obtained a very good result; Kirnberger temperament does the best in consonance than all the other temperaments except for just intonation, and Werckmeister wins on modulation. Note that they locate just near on the line linked by just intonation and

<sup>5</sup> <https://www.hooktheory.com/theorytab/>

<sup>6</sup> <https://insights.spotify.com/us/2015/05/06/most-popular-keys-on-spotify/>

	<i>SS</i>	<i>PT</i>	<i>JI</i>	<i>ET</i>	<i>QM</i>	<i>CQM</i>	<i>Wm.</i>	<i>Kb.</i>
<i>C</i>	0.9407	0.9478	0.8366	0.9276	0.8598	0.8598	0.8861	0.8668
<i>G</i>	0.9478	0.9478	0.8514	0.9276	0.8598	0.8598	0.9084	0.8463
<i>D</i>	0.9478	0.9250	0.9008	0.9276	0.9120	0.8598	0.9251	0.8744
<i>A</i>	0.9478	0.8969	0.9811	0.9276	0.9926	0.8598	0.9261	0.9161
<i>E</i>	0.9478	0.8521	1.0003	0.9276	1.0365	0.9120	0.9325	0.9443
<i>B</i>	0.9478	0.8678	1.0151	0.9276	1.0544	0.9926	0.9385	0.9417
<i>F</i> $\sharp$	0.9478	0.8956	0.9735	0.9276	1.0055	1.0365	0.9487	0.9435
<i>C</i> $\sharp$	0.9250	0.9407	0.9215	0.9276	0.9382	1.0544	0.9478	0.9453
<i>G</i> $\sharp$	0.8969	0.9478	0.8366	0.9276	0.8598	1.0055	0.9433	0.9485
<i>D</i> $\sharp$	0.8521	0.9478	0.8514	0.9276	0.8598	0.9382	0.9321	0.9478
<i>A</i> $\sharp$	0.8678	0.9478	0.8560	0.9276	0.8598	0.8598	0.9051	0.9380
<i>F</i>	0.8956	0.9478	0.8720	0.9276	0.8598	0.8598	0.8865	0.9096

Table 2: Average Consonant Level of Chords in Different Keys in Each Temperament

Mjor Keys	Parallel Keys	Total
<i>C</i>	10.20%	15.00%
<i>G</i>	10.70%	14.90%
<i>D</i>	8.70%	12.90%
<i>A</i>	6.10%	8.60%
<i>E</i>	3.60%	5.70%
<i>B</i>	2.60%	3.80%
<i>F</i> $\sharp$	2.70%	3.60%
<i>C</i> $\sharp$	6.00%	9.20%
<i>G</i> $\sharp$	4.30%	7.30%
<i>D</i> $\sharp$	2.40%	4.80%
<i>A</i> $\sharp$	3.50%	6.10%
<i>F</i>	5.30%	7.90%

Table 3: Usage of Keys in Popular Music

equal temperament, which implies that they are balanced between the two criteria.

Here, we have to note that this graph is biased by the usage of chords found in Spotify<sup>TM</sup> database, *i.e.*, the usage of chords are more inclined to that in the modern popular music. On the contrary, the mean-tone, dotted on the upper-right corner in the figure, was invented to obtain the clear resonance of the major third preferred in classicist age. It is easily guessed that if we employ the database of classical music the tendency would be different. The variety of distribution of dots in this space would surely reflect the difference of music genre, and this is our future work.

## 5. DISCUSSION AND CONCLUSION

We have proposed an index to show the numerical consonance level of triad, DIT, and have compared the difference of the level in various temperaments. Since chords on a scale may have different significance, we have weighted them by the number of appearance. The resultant difference has been visualized in various graphs.

Nowadays, we do not need to stick to the five-line staff based on 12-tone equal temperament in composing music

	Avg	SD
Sanfen Sunyi-fa	0.9245	0.0446
Pythagorean Tuning	0.9277	0.0425
Just Intonation	0.8981	0.0868
Equal Temperament	0.9276	0.0000
Quarter-Comma Meantone	0.9154	0.0917
Conventional QC Meantone	0.9267	0.0972
Werckmeister	0.9227	0.0284
Kirnberger	0.9120	0.0491

Table 4: DIT Results (SD is the standard deviation)

since actual performance should tolerate micro-tones, out-of-tune tones, portament, vibrating tunes, and so on. This tendency would be more salient in computer music age in future. It may be high time for us to reconsider traditional temperaments, to give special savors in music or to escape temporarily from the equal temperament, so that we should know the concrete difference in temperaments.

## Acknowledgments

This work is supported by JSPS kaken 16H01744.

## References

- [1] H. L. Helmholtz, *On the Sensations of Tone as a Physiological Basis for the Theory of Music*. Dover, 1954.
- [2] R. Plomp and W. J. M. Levelt, "Tonal consonance and critical bandwidth," *The journal of the Acoustical Society of America*, vol. 38, no. 4, pp. 548–560, 1965.
- [3] P. N. Vassilakis, "Perceptual and physical properties of amplitude fluctuation and their musical significance," Ph.D. dissertation, University of California, Los Angeles, 2001.

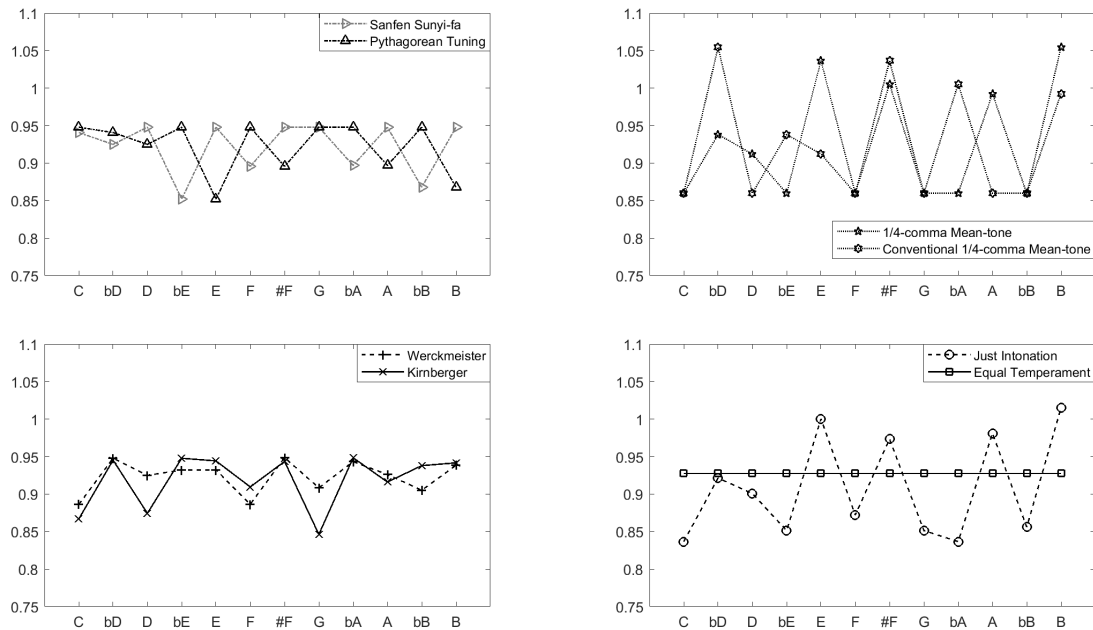


Figure 5: Comparison of Temperaments in Keys versus DIT Values; (upper-left) Sanfen Sunyi-fa vs Pythagorean, (upper-right) two mean-tones, (lower-left) two well temperaments, and (lower-right) the just intonation and the equal temperament

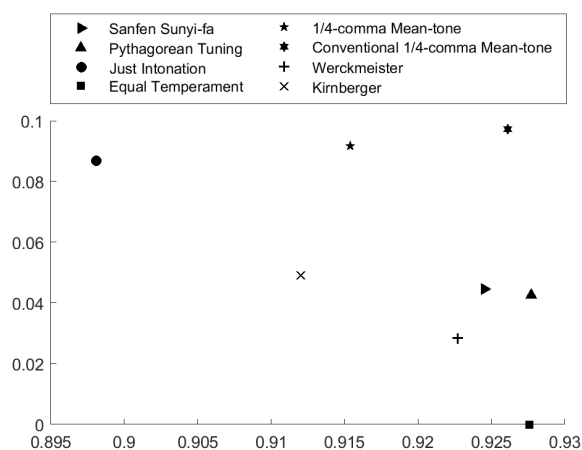


Figure 6: Consonance and Modulation Distribution

- [4] A. Ogata, *Science in Music Temperaments and Scales*. Kodan-sha blue backs (in Japanese), 2007.
- [5] N. D. Cook, "Harmony perception: Harmoniousness is more than the sum of interval consonance," *Music Perception: An Interdisciplinary Journal*, vol. 27, no. 1, pp. 25–42, 2009.
- [6] B. Benward and M. N. Saker, "Music: In theory and practice, seventh edition, 2 vols." Boston: McGraw-Hill, 2003, p. 56.
- [7] L. Tianquan, "Lun guanzi de sanfen sunyi-fa [talking about guanzi's sanfen sunyi-fa]," in *Yi Shu Tan Suo [Art Explore]*, 1995, pp. 62–65.
- [8] D. J. Benson, "Music: a mathematical offering," *The Mathematical Intelligencer*, vol. 30, no. 1, pp. 76–77, 2008.
- [9] G. Zarlino and V. Cohen, "On the modes: part four of le istitutioni harmoniche, 1558." Yale University Press, 1983.
- [10] F. d. Salinas, "De musica libri septem," vol. 11. Ediciones Universidad de Salamanca, 2014.
- [11] A. Werckmeister, "Musicae mathematicae hodegus curiosus." Georg Olms Verlag, 1972.
- [12] D. Ledbetter, *Bach's Well-tempered clavier: the 48 preludes and fugues*. Yale University Press, 2002.
- [13] A. Kameoka and M. Kuriyagawa, "Consonance theory part ii: Consonance of complex tones and its calculation method," *The Journal of the Acoustical Society of America*, vol. 45, no. 6, pp. 1460–1469, 1969.
- [14] W. Hutchinson and L. Knopoff, "The acoustic component of western consonance," *Journal of New Music Research*, vol. 7, no. 1, pp. 1–29, 1978.
- [15] W. A. Sethares, "Consonance-based spectral mappings," *Computer Music Journal*, vol. 22, no. 1, pp. 56–72, 1998.
- [16] G. Békésy, *Experiments in hearing*. McGraw-Hill, 1960.
- [17] E. Terhardt, "On the perception of periodic sound fluctuations (roughness)," *Acta Acustica united with Acustica*, vol. 30, no. 4, pp. 201–213, 1974.

# Composing space in the space: an Augmented and Virtual Reality sound spatialization system

Giovanni Santini

Hong Kong Baptist University  
info@giovannisantini.com

## ABSTRACT

This paper describes a tool for gesture-based control of sound spatialization in Augmented and Virtual Reality (AR and VR). While the increased precision and availability of sensors of any kind has made possible, in the last twenty years, the development of a considerable number of interfaces for sound spatialization control through gesture, their integration with VR and AR has not been fully explored yet. Such technologies provide an unprecedented level of interaction, immersivity and ease of use, by letting the user visualize and modify position, trajectory and behaviour of sound sources in 3D space. Like VR/AR painting programs, the application allows to draw lines that have the function of 3D automations for spatial motion. The system also stores information about movement speed and directionality of the sound source. Additionally, other parameters can be controlled from a virtual menu. The possibility to alternate AR and VR allows to switch between different environment (the actual space where the system is located or a virtual one). Virtual places can also be connected to different room parameters inside the spatialization algorithm.

## 1. INTRODUCTION AND BACKGROUND

Sound spatialization has been used as a resource for musical expression at least since Willaert's production at *Basilica di San Marco* in Venice (mid 16th century) [1]. More recently, since the first implementations of electronic music and especially in the past few decades, with the development of advanced sound spatialization algorithms (e.g., Vector-based Amplitude Panning (VBAP) [2], Higher Order Ambisonics (HOA) [3]), spatial sound has become a key element of the compositional syntax for an increasing number of composers: "space as a finality in music expression" (Leo Kupper in [4]) and "space as a compositional language" ([5]).

Since the first experiments by Pierre Schaeffer in the early 50s [1] one of the key aspects has been the control of the trajectories of sound sources (i.e., how to manipulate position coordinates through a "high-level" interface), along with the composition of many other parameters that can

affect sound perception (e.g. directivity, aperture of sound source and room characteristics).

Many solutions have been developed by providing some form of graphic editing/automations. In order to achieve intuitiveness and ease of use in a context where a big number of parameters comes into play, often some specific form of gestural input has been deployed. Gestural interfaces include tablets or gamepads ([6], [7]), gesture recognition through camera input, both for visible light and infrared ([8], [9]), or different sensors ([10]). More extensive reviews can be found in [11] and [12].

One further differentiation among systems can be identified between real-time sound spatialization systems or off-line studio editing applications: in the latter group can be inscribed systems responding to the needs of computer-aided composition, i.e. intuitive controls to be connected to the development of a musical structure ([13], [6]). Real-time control systems can often be referred to as DMI (Digital Musical Instrument [11], [14]) and more specifically as Spatialization Instruments, defined as "a Digital Musical Instrument, which has the capability of manipulating the spatial dimension of the produced sound, independently of its capability of producing or manipulating the other sound dimensions" [12].

Notwithstanding the high differentiation in functionalities and implementation details, all the cited input models result in some kind of symbolic representation that does not show the sound source in its exact position in space. In other words, none of those system lets the user see and control the sound trajectory "as it is". Overcoming such limitations might provide a better control, as "[...] devices whose control structures match the perceptual structure of the task will allow better user performances." ([15], referring to [16]).

In the case of Spatialization Instruments, "matching the perceptual structure of the task" would mean to exactly see where the sound source is positioned in space<sup>1</sup>.

The recent advancements in VR and AR technologies provide the background for representing the sound location.

## 2. DESCRIPTION OF THE SYSTEM

The described tool allows to represent and control the behaviour of sound sources in a 3D immersive space, as well as to edit other sound source parameters and store, save and

<sup>1</sup> The limitations of direction and distance perception (that would counteract the idea of clear identification of sound source position and trajectory) will be discussed later.



recall those data. Such automations can be modified after creation. Representation of positioning is in real-world scale and has a reduced level of abstraction, prioritizing as much as possible intuitiveness and matching visual objects to sound behaviour.

The Augmented Reality implementation allows to see and place sources in the real space. The VR mode provides interaction with virtual environments. Different (real and virtual) locations can be linked to different audio room settings inside the spatialization algorithm.

The system is developed through the interaction of two main components:

- an AR/VR project developed in *Unity3D* for the *HTC Vive Pro* headset;
- a Max/MSP patch dedicated to sound spatialization by using *Spat* (Ircam tools).

The two programs talk to each other through OSC (Open Sound Control) protocol.

The system has been tested in the *LIATe* (Lab for Immersive Arts and Technology) at Hong Kong Baptist University, with a 24.2 channels setup.



Figure 1. 10 sources distributed over the Sound Spatialization setup in the LIATe shown in the Max object *spat5.oper*.

## 2.1 The Unity Project

The AR session is implemented in Unity for HTC Vive Pro, currently the only headset allowing both VR and AR applications.

The input comes from the two controllers for the Vive, which have 6 DOF (Degrees Of Freedom) motion tracking.

The right controller allows the positioning of one sound source at a time through *parenting* (an operation by which a virtual object is linked in position and rotation to another object). By moving the controller and pressing the back trigger, the user can create/modify the trajectory of the selected sound source. Such trajectory is shown as a line drawn in the air. As a child<sup>2</sup>, a source can be given an offset respect to the parent controller, thus translating and magnifying the movement of the controller (for example, by shifting the sound source one meter above the controller on the Y axis, a 360 rotation of the controller would create a 2m diameter circle centered on the controller).

<sup>2</sup> A parent is the object providing the reference coordinate system, while a child is a virtual object whose coordinates are referred to the coordinates of the parent.



Figure 2. Point of view 1 on a combination of sources and trajectories.

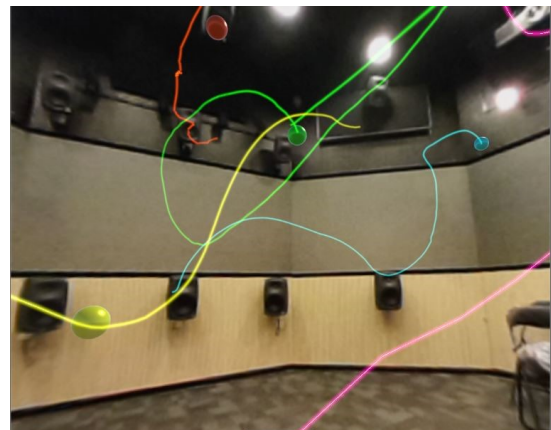


Figure 3. Point of view 2 on the same combination.

The position of sound sources (update frame by frame) is sent through OSC to Max/MSP (that performs the sound spatialization).

In the current state of development, the application allows to control up to 10 sound sources at the same time.

The left controller can move an additional sound source. Furthermore, it has a User Interface (UI) attached allowing for the selection of different tools (the UI only sends OSC commands to Max/MSP, which actually performs the tasks):

- shifting the sound source from the parent controller (over the three different axes);
- selecting and soloing (if needed) different sound sources and assigning different trajectories (recognizable by different colors);
- changing the aperture and yaw of the selected source;
- choosing the spatialization algorithm;
- changing the room (as a VR room);
- storing and recalling those trajectories; changing trajectories after drawing.



Figure 4. The Menu attached to the controller.

Sound sources are visualized as spheres of different colors; when they move, either they follow a trajectory or are moved by a controller. The trajectory is not followed with a fixed speed: speed is changed according to the original gesture (every trajectory is, originally, drawn with a gesture). If the sound source's duration is longer than the trajectory's one (e.g., the sound source is 3 seconds and the trajectory is 2 seconds long), the sound source is left static on the last point of the trajectory. However the gesture representation can be always edited in real time by pressing the trigger of the controller. Thus, the user can freely adjust a trajectory to the sound source it is related to.

## 2.2 Sound spatialization

OSC bundles sent out from Unity are received by a Max/MSP patch based on *Spat* (Ircam tools). As both Unity and Spat use a coordinate system where 1 corresponds to 1 meter, the passage from one system to the other does not require remapping except for coordinate systems alignment. While the AR/VR project in Unity can be considered the front-end of the application, all the core functions are actually implemented in Max/MSP and most of the functions control Spat parameters (position, sound source aperture, yaw, etc.).

The system uses different "coll" objects (each one for every different sound source), in order to store, save and recall trajectory information.

Different spatialization algorithms are available (e.g. 3D VBAP, HOA and binaural) [17], and their use is left to the discretion of the user.

Sources moving along trajectories can also be saved as audio tracks.

## 3. LOCALIZATION OF SOUND AND VIRTUAL OBJECTS

The presented application is based on a relation between virtual object position and sound source position; therefore a critical issue must be considered: distance estimation and correspondence of visual and aural movements.

As [18] shows, the vision-based distance estimation of a virtual object presents problems in an AR environment. While the angular positioning is rather precise, the understanding of distance tends to be underestimated. The

study evaluates numerous rendering strategies for virtual objects (such as aerial perspective<sup>3</sup>, cast shadows<sup>4</sup> and shading<sup>5</sup>). The authors find, through two specifically designed experiments, that the most effective (by far) rendering strategy to reduce the underestimation of distance consists of casting shadows on the floor (rendered shadows are created by a virtual source of light perpendicular to the floor). In fact, in both experiments, cast shadows proved to increase accuracy in distance estimation respectively by 90% and 18%.

For audio discrimination, as shown in [19] and [20] many parameters and spectral cues enter into play: sound level, direct-to-reverberant ratio (DRR), spectral shape (e.g., low-pass filtering of frequencies in function of the distance), binaural cues like Interaural Time Differences (ITDs) and Interaural Level Differences (ILDs), dynamic cues (motion) and familiarity with the sound. Even though such cues are important for giving an idea of distance, a precise estimation of the perceived distance is problematic. In fact, given the complexity of the overall perceptual system and the dependency of recognition upon many different factors, including the conformation of the venue itself, distance perception is biased and tends to underestimation.

[19] also shows that the presence of a visual cue can help in focusing the position of a sound source (sometimes producing *ventriloquism*, the phenomenon that occurs when a listener mistakenly adjusts the perception of sound localization to the position of the visual cue).

Moreover, the discrimination of behaviour of sources is made problematic by some other effects: for instance, one sound tends to be more sharply localized when its position coincides with the one of a real speaker. Another phenomenon we can take as an example, named as *flickering* in [5], consists in the impossibility for our hearing to discriminate position under a very fast source movement, or better, the tendency to ignore most part of a trajectory, by focusing only on some discontinuous points in space.

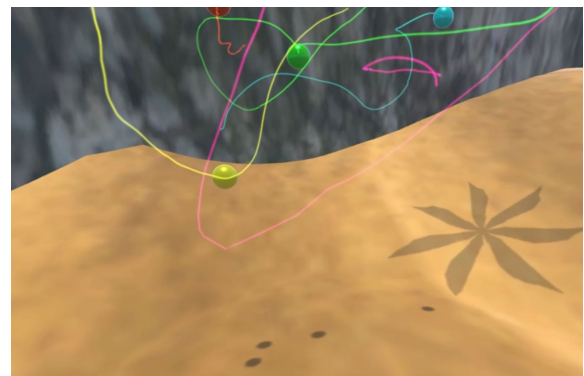


Figure 5. The same configuration of Figure 2 and 3 but in VR (and with down-cast shadows).

According to [19] and [20] the simultaneous presence of both visual and aural cues helps in discriminating position,

<sup>3</sup> Increased haziness of colors with the increase of distance.

<sup>4</sup> Renderings of virtual shadows on the floor.

<sup>5</sup> Defining the reflectance properties of a virtual object.

distance and behaviour of a sound source; as down-cast shadows<sup>6</sup> help to have a correct estimation of virtual objects position, they further increase the precision of sources localization.

#### 4. USES, LIMITATIONS AND FUTURE WORKS

The presented tool allows to control sound spatialization in an immersive environment, providing the visualization of sound sources' positions and trajectories. It allows fast testing of spatial compositional solutions and real-time control over numerous spatialization parameters. It can be used live as a Spatialization Instrument or off-line as a sort of (limited) Digital Audio Workstation (DAW).

As pointed out in [6] the limit of some gesture-controlled (real-time) systems might fall short for what concerns large-scale conception and compositional organization, especially in relation to musical structures that might prescind from bodily gestures. For this reason, a future improvement should include the possibility to edit trajectories even in a computer-aided composition context.

The Spatial Instrument described might seem to follow from a *naïve* approach: sound trajectories can be perceived with the same clarity of our visual perception (i.e., the two representations, visual and aural, of a movement are, to some extent, precise and identical). As already shown in [5], [19], [20] even hearing under the most ideal conditions, perceived distances appears to be “a biased estimate of physical source distance” [19]. As the perception of distance (but also of behaviour over time) is influenced by spectral characteristics of sound, the proposed system can be useful as a way for “fast prototyping”, but cannot solve the problems inherent to sound spatialization, that in numerous cases require a tailored approach to different sound sources, sound fields and timbres.

In addition to the source-trajectory approach shown in this paper, another resource might be found in a spectral spatialization approach. One possible idea would consist of distributing different frequency bands of one audio file across the space as if they were different sound sources and providing each band with dynamic movements; while such approach could not have a single-bin accuracy while maintaining intuitiveness of use, bin grouping based on psychoacoustic perception (such as Bark bands [21]) would certainly be possible. Therefore, it would be possible to obtain a fluctuating timbral environment by organizing the movement of different Bark bands inside one timbre.

Moreover, a future study will be addressed to the assessment of the usability and usefulness of the tool both with trained musicians and untrained people.

#### 5. CONCLUSION

The paper has described a VR/AR immersive system for sound spatialization. It allows real-time control over position, trajectory and other parameters of different sound

sources, visualized as spheres. Trajectories are visualized as virtual strokes.

The Digital Instrument mapping is intuitive, as sounds' positions and trajectories mirror the gesture of the player. These gestures can be translated in space and scaled (a small movement can result in a shift of several meters). A simple UI attached to the left controller allows the user to change different parameters and options (spatialization algorithm, sound source, aperture and yaw etc.). The application can be also used as a tool for automating trajectories and can be useful for electroacoustic composition. Data about sources movements can be stored as text in “coll” objects; spatialized soundfiles can also be exported as audiofiles.

The switch from AR to VR changes the environment where virtual sources are visualized from the real world to a VR landscape. Such possibility to switch makes it easier to render on the floor shadows of virtual objects representing sound sources. As [18] shows, such shadows, rendered under the objects with a virtual light perpendicular to the floor, increase the accuracy of estimation of virtual objects positions.

The intuitiveness of the system is enhanced by the simultaneous presence of both visual (representation of sound sources and trajectories) and aural cues. On the other side, such close mimicking between sound and visual behaviour might induce a simplistic approach (as if localization of sound sources could always be perfectly accurate). The user should always consider some degree of inaccuracy due to intrinsic characteristic of sound spatialization: the understanding of source positioning is influenced by many parameters, such as intensity, direct-to-reverb ratio, and spectral EQ. Consequently, in numerous circumstances, a case by case approach should be considered.

#### 6. REFERENCES

- [1] R. Zvonar, “A history of spatial music,” *CEC*, 1999.
- [2] V. Pulkki, “Virtual Sound Source Positioning Using Vector Base Amplitude Panning,” *Audio Engineering Society*, 1997.
- [3] D. Malham, “Higher order ambisonic systems for the spatialisation of sound,” in *1999 International Computer Music Conference (ICMC)*, 1999.
- [4] R. Normandeau, “Timbre spatialisation: The medium is the space,” 2009.
- [5] T. Schmele, “Exploring 3D Audio as a New Musical Language,” Master's Thesis, Universitat Pompeu Fabra, 2011.
- [6] J. Garcia, J. Bresson, and T. Carpentier, “Towards interactive authoring tools for composing spatialization,” in *2015 IEEE Symposium on 3D User Interfaces, 3DUI 2015 - Proceedings*, 2015.
- [7] K. Bredies, N. A. Mann, J. Ahrens, M. Geier, S. Spors, and M. Nischt, “The multi-touch SoundScape renderer,” in *Proceedings of the working conference on Advanced visual interfaces - AVI '08*, 2008.

<sup>6</sup> Down-casting shadows in AR requires a 3D scanning of the environment. HTC Pro has the capability to do so, but the range is rather limited and subject to visual artifacts. In VR shadows are easy to represent properly.

- [8] D. Copeland, "The NAISA Spatialization System," 2014. [Online]. Available: <http://www.darrencopeland.net/web2/?page{ }id=400>
- [9] W. Fohl and M. Nogalski, "A Gesture Control Interface for a Wave Field Synthesis System," in *Nime 2013 Proceedings of the International Conference on New Interfaces for Musical Expression*, 2013.
- [10] M. L. Hedges, "An investigation into the use of intuitive control interfaces and distributed processing for enhanced three dimensional sound localization," Master thesis, Rhodes University, 2015.
- [11] A. Pysiewicz and S. Weinzier, "Instruments for Spatial Sound Control in Real Time Music Performances. A Review." in *Musical Instruments in the 21st Century*. Singapore: Springer, 2017, pp. 273–296.
- [12] A. Pérez-Lopez, "Real-Time 3D Audio Spatialization Tools for Interactive Performance," *Universitat Pompeu Fabra, Barcelona*, p. 38, 2014.
- [13] R. Gottfried, "SVG to OSC transcoding as a platform for notational Praxis and electronic performance," in *Proceedings of the International Conference on Technologies for Music Notation and Representation*, Paris, 2015, pp. 154–161.
- [14] J. Malloch, D. Birnbaum, E. Sinyor, and M. M. Wanderley, "Towards a new conceptual framework for digital musical instruments," in *Proceedings of the 9th International Conference on Digital Audio Effects*, 2006.
- [15] M. M. Wanderley and N. Orio, "Evaluation of input devices for musical expression: Borrowing tools from HCI," *Computer Music Journal*, 2002.
- [16] R. J. K. Jacob, L. E. Sibert, D. C. McFarlane, and M. P. Mullen, Jr., "Integrity and separability of input devices," *ACM Trans. Comput.-Hum. Interact.*, vol. 1, no. 1, pp. 3–26, Mar. 1994. [Online]. Available: <http://doi.acm.org/10.1145/174630.174631>
- [17] T. Carpentier, M. Noisternig, and O. Warusfel, "Twenty years of Ircam Spat: looking back, looking forward," *International Computer Music Conference Proceedings*, 2015.
- [18] C. Diaz, M. Walker, D. A. Szafr, and D. Szafr, "Designing for depth perceptions in augmented reality," in *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Oct 2017, pp. 111–122.
- [19] P. Zahorik, D. S. Brungart, and A. W. Bronkhorst, "Auditory distance perception in humans: A summary of past and present research," *Acta Acustica united with Acustica*, 2005.
- [20] A. J. Kolarik, B. C. Moore, P. Zahorik, S. Cirstea, and S. Pardhan, "Auditory distance perception in humans: a review of cues, development, neuronal bases, and effects of sensory loss," *Attention, Perception, and Psychophysics*, 2016.
- [21] E. Zwicker, "Subdivision of the audible frequency range into critical bands (frequenzgruppen)," *The Journal of the Acoustical Society of America*, vol. 33, no. 2, pp. 248–248, 1961.



# Graph Based Physical Models for Sound Synthesis

**Pelle Juul Christensen**  
Aalborg University  
Copenhagen, Denmark  
pelle.juul@tuta.io

**Stefania Serafin**  
Multisensory Experience Lab  
Aalborg University Copenhagen  
Copenhagen, Denmark  
sts@create.aau.dk

## ABSTRACT

We focus on physical models in which multiple strings are connected via junctions to form graphs. Starting with the case of the 1D wave equation, we show how to extend it to a string branching into two other strings, and from there how to build complex cyclic and acyclic graphs. We introduce the concept of dense models and show that a discretization of the 2D wave equation can be built using our methods, and that there are more efficient ways of modelling 2D wave propagation than a rectangular grid. We discuss how to apply Dirichlet and Neumann boundary conditions to a graph model, and show how to compute the frequency content of a graph using common methods. We then prove general lower and upper bounds computational complexity. Lastly, we show how to extend our results to other kinds of acoustical objects, such as linear bars, and how to add dampening to a graph model. A reference implementation in MATLAB and an interactive JUCE/C++ application is available online.

## 1. INTRODUCTION

Recent research in physical models has been directed towards simulating systems using *finite difference schemes* [1], which can be used to model the intricacies of many kinds of vibrating systems and excitors.

As many other physical modelling methods, finite difference schemes can be used to simulate systems which are hard to construct in real life. For example, we can tweak the parameters of models to make e.g., strings that are extremely long or violin bows that move faster than the human anatomy allows for. Some research has explored this idea further by building abstract physical models that do not have any direct relation to any real world instruments. For example, the work of Stefan Bilbao includes ways of constructing modular percussion instruments by connecting vibrating bars and plates [2]. Similarly, the CORDIS-ANIMA project of ACROE allows one to build virtual instruments by combining masses, springs, friction elements and non-linear links, to create novel compositions and matching animations [3].

Copyright: © 2019 Pelle Juul Christensen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

This paper will further explore the notions of abstract physical models by showing a way to build systems of connected strings that would not necessarily be achievable or practical in real life. The models described in this paper could possibly be constructed physically; the goal of the current study is, however, to build synthesis algorithms inspired by physical phenomena but without attention to whether they are realisable in real life or not.

While we are looking at finite difference schemes we should be aware that many methods of physical modelling for sound synthesis exists and that they are largely equivalent with regards to which sounds we can produce using them. Related to the work at hand are waveguides, modal synthesis and mass-spring systems [4] [5], [6]. For a nice description and discussion of the various methods see [1, Chapter 1].

We start in Section 2 by reviewing the one-dimensional wave equation and how to make a finite-difference scheme to simulate it. In Section 3 we take a derivation of the regular 1D wave equation, and show how to get similar results for a branching topology. In Section 4 we see how to use these results to build various kinds of models. In Section 5 we investigate the properties of *dense models*, and show that a special case of these is equivalent to a discretization the 2D wave equation. In section 6 we deal with the boundary conditions of our models. Next, in Section 7 we discuss the computational complexity of various models, and in Section 8 we give a method of computing natural mode frequencies and shapes. Lastly in Sections 10, 11 and 12 we end with a description of the reference implementation, suggestions for future work, and concluding remarks.

## 2. THE 1D WAVE EQUATION

Before building complex abstract models, we will review the 1D wave equation, which will be used as the starting point for further exploration, since it is thoroughly studied and perhaps the simplest spatial model of musical utility. This section, provided for completeness, is completely textbook and based on [2], [1], [7], [8], and [9].

The 1D wave equation is defined by the second-order partial differential equation

$$u_{tt} = c^2 u_{xx}, \quad (1)$$

where  $u = u(x, t)$  is a variable describing the deformation of the medium at position  $x \in [0, 1]$  and time  $t \in [0, \infty]$ . The constant  $c$  is the normalized wave speed which is determined by the medium under consideration. When dis-

cretized, Equation (1) looks like

$$\delta_{tt}u_l^n = c^2\delta_{xx}u_l^n, \quad (2)$$

where the finite difference operators  $\delta_{tt}$  and  $\delta_{xx}$  are defined as

$$\delta_{tt}u_l^n = \frac{1}{k^2}(u_l^{n+1} - 2u_l^n + u_l^{n-1}) \approx \frac{d^2u}{dt^2}, \quad (3)$$

$$\delta_{xx}u_l^n = \frac{1}{h^2}(u_{l+1}^n - 2u_l^n + u_{l-1}^n) \approx \frac{d^2u}{dx^2}. \quad (4)$$

To implement this model, we expand the temporal operator in Equation (2) and isolate  $u_l^{n+1}$  to get

$$u_l^{n+1} = k^2c^2\delta_{xx}u_l^n + 2u_l^n - u_l^{n-1}, \quad (5)$$

### 3. TWO-BRANCH AND N-BRANCH TOPOLOGIES

The 1D wave equation can be considered an idealization of a real physical string (e.g. a guitar string) under low amplitude conditions. We can represent this using a graph with two nodes and one edge, as seen in Figure. 1. a).

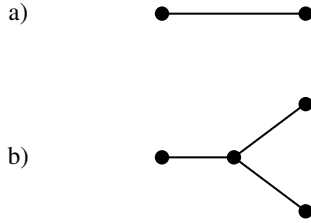


Figure 1. a) A line topology model — a string viewed as a graph with two nodes and one edge. b) A branching topology model — one string branching out into two other strings.

Once we look at our system as a graph, a new perspective arises: if we can build this kind of graph, what other graphs can we create? In this section we will look at the case of a *branching* topology — one string segment connected to two other string segments through one node, as visualized in Figure 1. b). We will investigate how to model wave propagation on such a graph.

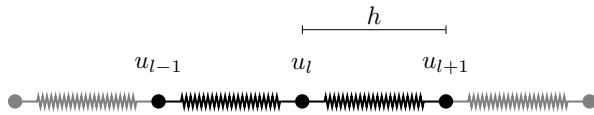


Figure 2. A section of a line topology string view as a lumped mass-spring network.

To understand the branching topology we must first look at the line topology more in-depth. We will be using a derivation found in [1, Chapter 6] and similarly in [8]. In Figure 2. we see a section of a string viewed as a network of masses connected via springs. The dynamics of the mass  $u_l$  will be defined by the ordinary differential equation

$$m\frac{d^2u_l}{dt^2} = f_{l+1,l} - f_{l,l-1}, \quad (6)$$

where  $m$  is the mass of the node and e.g.  $f_{l+1,l}$  is defined by

$$f_{l+1,l} = \kappa(u_{l+1} - u_l), \quad (7)$$

which is the force caused by the spring between  $u_{l+1}$  and  $u_l$ , where  $\kappa$  is the spring constant.

Combining Equations (6) and (7) we get

$$m\frac{d^2u_l}{dt^2} = \kappa(u_{l+1} - 2u_l + u_{l-1}). \quad (8)$$

Defining  $m = \rho Ah$  where  $\rho$  is density,  $A$  the cross-sectional area of the string, and  $h$  is the distance between the node. Then setting  $\kappa = EA/h$ , where  $E$  is the Young's modulus of the material, we get

$$\frac{d^2u_l}{dt^2} = \frac{E}{\rho} \left( \frac{u_{l+1} - 2u_l + u_{l-1}}{h^2} \right). \quad (9)$$

Notice that the right-hand side of this equation is equivalent to  $c\delta_{xx}u$  when  $c = \sqrt{E/\rho}$ .

Now we perform the same derivation, but for the branching topology. A mass-spring network of the branching point is shown in Figure 3. For simplicity and for the remainder of this paper we will assume that all connected strings has the same parameters ( $h$ ,  $k$  and  $c$ ).

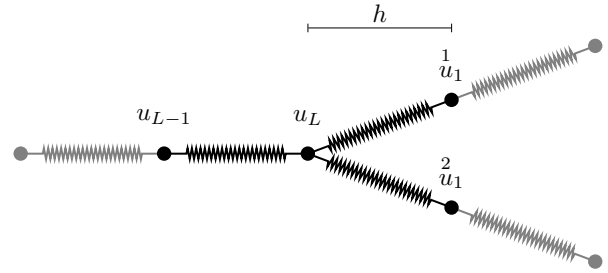


Figure 3. The branching point of a branching topology viewed as a mass-spring network.

The dynamics of  $u_L$  is described by

$$m\frac{d^2u_L}{dt^2} = f_{1,L}^1 + f_{1,L}^2 - f_{L,L-1}. \quad (10)$$

Notice that we are using  $L$  instead of  $l$  since we are at the end of string segment  $u$  and that we are using 1 instead of  $l$  for  $u$  since we are at the beginning of those string segments.

The spring forces of the branching node is

$$f_{1,L}^1 = \kappa(u_1^1 - u_L), \quad (11)$$

$$f_{1,L}^2 = \kappa(u_1^2 - u_L). \quad (12)$$

Taking the same steps as we did to reach Equation (9) but using Equations (10) through (12) we get

$$\frac{d^2u_L}{dt^2} = \frac{E}{\rho} \left( \frac{u_1^1 + u_1^2 + u_{L-1} - 3u_L}{h^2} \right). \quad (13)$$

By analogy to Equation (9) we then have a definition for  $\delta_{xx}u_L$  in the case of the branching topology, which is

$$\delta_{xx}u_L = \frac{1}{h^2}(u_1 + u_1 + u_{L-1} - 3u_L). \quad (14)$$

Of course we are not limited to just two branches. We can create a N-branch topology as in Figure 4.

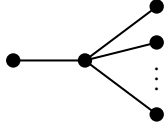


Figure 4. A model with an  $N$ -branch topology.

Using a derivation similar to that of the two-branch model, we get a definition of  $\delta_{xx}u_L$  which looks like

$$\delta_{xx}u_L = \frac{1}{h^2} \left( \sum_{u \in \mathbb{U}_L} u^* - |\mathbb{U}_L|u_L \right), \quad (15)$$

where  $\mathbb{U}_L$  is the set of all the end nodes of the string segments connected to the branching node  $u$  including  $u_{L-1}$ .

For now we have only considered a branch at the end of a string. We could just as well have considered a branch at the beginning and arrived at a result similar to Equation (15). We can generalize our rule such that we can apply that rule to any point in our graph, even the internal nodes:

$$\delta_{\Delta\mathbb{U}}u_l = \frac{1}{h^2} \left( \sum_{u \in \mathbb{U}_l} u^* - |\mathbb{U}_l|u_l \right). \quad (16)$$

Notice that we have changed our notation to  $\delta_{\Delta\mathbb{U}}$  instead of  $\delta_{xx}$ . This is to avoid confusion with the one-dimensional  $\delta_{xx}$  and the  $|\mathbb{U}_l|$  dimensional  $\delta_{\Delta\mathbb{U}}$ .

Note that the angle between the strings in a junction are not considered since the nodes have no freedom of movement in the 2D plane in which we are building our graphs. Also, when drawing a graph we will usually not care about getting the distances and angles right, what's important is how the strings connect and how many internal nodes they have.

The notion of acoustic junctions is not a new concept in physical modelling. 2D and 3D finite difference schemes already contains scattering junctions arranged as grid [1, Chapter 6] [10], however the author has not seen the concept presented as in the current paper where the junctions do not need to be distributed homogeneously throughout the model. Likewise, the physical modeling method of *digital waveguides*, which has been proved equivalent to finite difference schemes [11], use scattering junctions to great extend in order to model room and instrument acoustics [12] [13].

#### 4. BUILDING MODELS

Using Equation (16) we are able to connect any number of strings together. For example we could take a topology

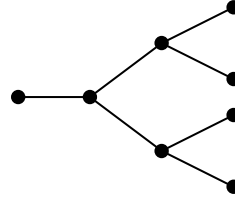


Figure 5. A model with a binary tree-like topology.

like Figure 1. b) and connect another layer to the rightmost nodes to get a topology like Figure 5.

In graph theory, we would call such a graph a *tree*. More generally we can call it an *acyclic graph*. Any graph we can build by only adding strings to the end of other strings will be a tree. So far all models we have looked have been trees.

We can build more exciting graphs by connecting the ends of two strings using another string. For example, we can take the graph in Figure 5 and connect the top right node to the leftmost node, resulting in the graph in Figure 6, which is a *cyclic graph*.

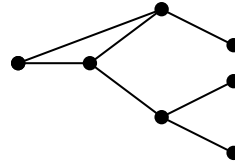


Figure 6. A graph with a single cycle

We may also take a string and connect its two ends, thus creating a loop as seen in Figure 7. This is a compelling case because it is a model that does not need any boundary conditions. In general, any graph without pendant nodes can be evaluated without boundary conditions. Physical models with looping topology has previously been studied in the case of Tibetan singing bowls and glass harmonicas [14].



Figure 7. A string with its two ends joined, forming a graph with a single cycle

#### 5. DENSE TOPOLOGIES

A point that has been implicit so far is that each string in a given model must be assigned a number of internal nodes, just like we assign the ordinary 1D wave equation number of nodes when we discretize it.

If we construct a model in which there are no strings longer than one between each branching node, we say that the model is dense<sup>1</sup>.

<sup>1</sup> Note that some dense models may have string segments longer than one at the edges, we will ignore this fact for now since the internal structure of such a model will be dense

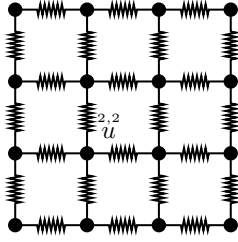


Figure 8. A dense rectangular grid. The zigzag lines, symbolising springs, are drawn to show that the model has no internal strings.

One well known dense model is the rectangular grid, as seen in Figure 8. When we wish to update one of the inner nodes  $^{x,y}u$  we use Equation (16), which in this case takes the form

$$\delta_{xx}^{x,y} u = \frac{1}{h^2} \left( ^{x,y-1}u + ^{x-1,y}u + ^{x+1,y}u + ^{x,y+1}u - 4^{x,y}u \right). \quad (17)$$

The two-dimensional version of the wave equation, which models wave propagation on a non-stiff membrane is defined by [1, Chapter 5]

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right). \quad (18)$$

A finite difference scheme for this could look like

$$\delta_{tt}^{x,y} u = c^2 \left( \delta_{xx}^{x,y} u + \delta_{yy}^{x,y} u \right). \quad (19)$$

If we expand the spatial difference operators we get

$$\delta_{xx}^{x,y} u + \delta_{yy}^{x,y} u = \frac{1}{h^2} \left( ^{x,y-1}u + ^{x-1,y}u + ^{x+1,y}u + ^{x,y+1}u - 4^{x,y}u \right), \quad (20)$$

which is equivalent to Equation (17). Therefore, the model in Figure 8. is equivalent to the discretized 2D wave equation.

However, since we are building grids using nodes and not from the definition of the 2D partial derivative, we can build grids which are not rectangular. For example, Figure 9. shows a hexagonal grid.

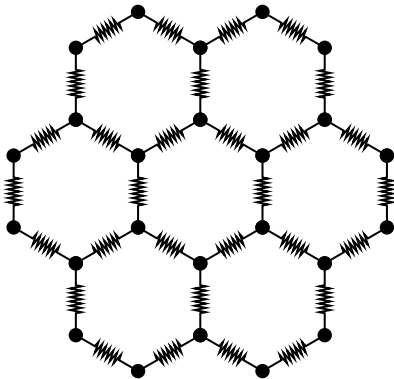


Figure 9. A dense hexagon grid.

Grid configurations are common throughout physical modeling literature. Especially the method of using digital wave-

guides has investigated various grid configurations and provide tools for building arbitrary typologies. Since Digital waveguides has as been proved equal to finite-difference schemes the result of those studies should be applicable to graph physical models as well [15] [5] [11].

Say we wanted to model the wave propagation on a 2D membrane, which kind of grid would be more efficient with respect to area covered? If we first look at the rectangular grid we see that we cover

$$R_4 = \frac{A}{N} = \frac{1}{4} \quad (21)$$

unit area per node. Where  $A$  is the area of a grid unit, and  $N$  is the number of nodes in each unit, considering only inner grid units.

If we do the same for the hexagonal grid we get

$$R_6 = \frac{A}{N} = \frac{3\sqrt{3}(1^2)/2}{6} = \frac{\sqrt{3}}{4}. \quad (22)$$

Since we have that

$$\frac{\sqrt{3}}{4} > \frac{1}{4}, \quad (23)$$

we can cover more area using the same amount of nodes when arranging them in a hexagonal grid, compared to a rectangular grid, giving us a more efficient model.

Furthermore, each node of an inner grid unit in the rectangular grid has connectivity 4 (each node connects to four other nodes), while the hexagon grid has connectivity 3. One can infer from the results of Section 7. that if we have the same amount of nodes, this causes the hexagon grid to have a lower computational complexity than the rectangular grid.

The differences between rectangular and hexagonal grids have been investigated before by Bilbao and Hamilton in [10], where they arrive at a conclusion similar the one presented here with the addition of also analysing the dispersion of each configuration.

Much is still left to be said about dense configurations. One does not have limit oneself to regular grids or grids at all. However, the nuances of various dense configurations are beyond the scope of the current project and could be enough work for a separate paper.

## 6. BOUNDARY CONDITIONS

To evaluate most models we need to decide on which boundary conditions to use for at least the pendant nodes.

When considering a string topology model we need only apply boundary conditions at the two ends. However, since graph models can have multiple pendant nodes, we might need to decide on more than two boundary conditions. In the case of models with no pendant nodes, we do not necessarily need any boundary conditions.

A non-pendant node may have multiple boundary conditions. Take the example of a rectangular plate which is clamped at the left and right edges and free to move at the top and bottom edges. In this case the corners of the plate will have a "free" boundary condition in the top-bottom direction and a "clamped" boundary condition in the left-right direction. For our graph models we will similarly



have to decide on a boundary condition for each direction, however, our models may have more than two directions. If a node in our model has boundary conditions we call it an *edge node*.

The number of boundary conditions to be decided for an edge node will be the same as its connectivity. For example, if we consider the middle node in Figure 1. b) to be an edge node, we need to implement boundary conditions for the directions of each of the three connected strings.

Two commonly used boundary conditions are [1]

$$u = 0 \quad (\text{Dirichlet}), \quad (24)$$

$$u_x = 0 \quad (\text{Neumann}). \quad (25)$$

For simplicity we will assume for a given node, all directions will have the same boundary condition.

To implement the Dirichlet condition we introduce a virtual node with a constant value 0 for each direction. This gives us the update rule

$$\delta_{\Delta U} u_l^n = \frac{1}{h^2} \left( \sum_{\tilde{u} \in \mathbb{U}_l} \tilde{u}^* - 2(|\mathbb{U}_l|) u_l \right). \quad (26)$$

Implementation of the Neumann condition involves setting  $\delta_{\tilde{u}} = \frac{1}{2h} (u_{l+1}^n - u_{l-1}^n) = 0$  for each direction, which amounts to creating a virtual node with the same value as the real node of that direction, see [1, chapter 5.2.8] for further details. Doing this we get an update rule which looks like

$$\delta_{\Delta U} u_l^n = \frac{1}{h^2} \left( 2 \sum_{\tilde{u} \in \mathbb{U}_l} \tilde{u}^* - 2|\mathbb{U}_l| u_l \right). \quad (27)$$

There are lots of other options for boundary conditions apart from Neumann and Dirichlet, see e.g. [9, chapter 19] for some choices.

## 7. COMPUTATIONAL COMPLEXITY

Knowing the computational complexity of a given model is critical if one wishes to run large models or run models in real time. Since graph theory is a well studied area of computer science a lot of existing material will cover complexity analysis similar to the one of this section (see e.g. [16]). Despite this we will still go through a basic complexity analysis specific to the topic at hand.

The complexity of a node will depend on the amount of connections to it. Evaluating Equation (16) for a pendant node will take just one operation when disregarding the division by  $h^2$ . Connected nodes will take  $|\mathbb{U}| + 1$  operations:  $|\mathbb{U} - 1|$  operations for the summation and two for the last addition and multiplication.

If we disallow cycles, the worst case model is then the string topology model because it has the lowest amount of pendant nodes. The complexity of the string model is  $3(n - 2) + 2 = \mathcal{O}(n)$ . The acyclic model with the highest amount of pendant nodes has one node connected to every other node in the model, this has complexity  $n + (n - 1) = \mathcal{O}(n)$ . Thus any acyclic will have complexity  $\mathcal{O}(n)$ .

If we allow cycles we may increase the complexity of our model. The most complex model is the one where every

node connects to every other node, giving us a complexity of  $n(n + 1) = \mathcal{O}(n^2)$ , which is thus the upper bound of any model we can build.

## 8. HARMONIC CONTENT

Finding the harmonic content of a given model can be done using common methods for computing the natural modes of a mass-spring system [1, chapter 3]. This method is completely standard and should be found in any good textbook on the subject, but is presented here specific to the topic at hand.

A given mass spring system can be described using the equation

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{0}, \quad (28)$$

where  $\mathbf{M}$  encodes the masses,  $\mathbf{K}$  the spring constants and  $\mathbf{u}$  the displacements of the masses.

We then assume that Equation (28) has a solution of the form

$$\mathbf{u} = \mathbf{U}e^{i\omega t}, \quad (29)$$

which we plug back into Equation (28) to get

$$(\mathbf{K} - \omega^2 \mathbf{M}) \mathbf{U} e^{i\omega t} = \mathbf{0} \quad (30)$$

and since  $e^{i\omega t} \neq 0$  we have that

$$(\mathbf{K} - \omega^2 \mathbf{M}) \mathbf{U} = \mathbf{0}. \quad (31)$$

Multiplying through by  $\mathbf{M}^{-1}$  we get

$$(\mathbf{M}^{-1} \mathbf{K} - \omega^2 \mathbf{I}) \mathbf{U} = \mathbf{0}, \quad (32)$$

which is analogous to the canonical form of the eigenvalue problem

$$(\mathbf{A} - \lambda \mathbf{I}) \mathbf{v} = \mathbf{0} \quad (33)$$

when setting  $\mathbf{A} = \mathbf{M}^{-1} \mathbf{K}$  and  $\lambda = \omega^2$ .

Therefore finding the eigenvalues of  $\mathbf{M}^{-1} \mathbf{K}$  will give us the frequencies of the natural modes of the system, the corresponding eigenvectors  $\mathbf{v}$  will be the shape of the given mode.

Since our models are characterized by the wave speed  $c^2$  and the topology of the graph, we need a way deriving  $\mathbf{M}$  and  $\mathbf{K}$  from these.

Using the definitions from Section 3. and selecting  $A = 1$  and  $\rho = 1$  we get

$$\kappa = \frac{c^2}{h} \quad \text{and} \quad m = h. \quad (34)$$

Since all nodes in our system have the same mass we have

$$\mathbf{M} = h \mathbf{I}. \quad (35)$$

The shape of  $\mathbf{K}$  will depend on the topology of the graph. For example, considering only the center node  $u_L$  of Figure 3. we get matrices which look like

$$\mathbf{K}\mathbf{u} = \begin{bmatrix} \kappa & -3\kappa & \kappa & \kappa \\ & \dots & & \\ & \dots & \dots & \\ & & \dots & \dots \end{bmatrix} \begin{bmatrix} u_{L-1} \\ u_L \\ \hat{u}_1 \\ \hat{u}_1 \end{bmatrix}. \quad (36)$$

This process of building the  $\mathbf{K}$  matrix can and should be done using software for models of large sizes.

## 9. EXTENSIONS

So far we have looked only at the case of building models from the 1D wave equation. It is, however, possible to use the same principles for other acoustic objects.

For example, we can take a linear bar model [1]

$$\frac{d^2}{dt^2}u = -\kappa^2 \frac{d^4}{dx^4}u, \quad (37)$$

which can be discretized as

$$\delta_{tt}u_l^n = -\kappa^2 \delta_{xx} \delta_{xx} u_l^n, \quad (38)$$

after which we apply Equation (10) instead of  $\delta_{xx}$  to get

$$\delta_{tt}u_l^n = -\kappa^2 \delta_{\Delta U} \delta_{\Delta U} u_l^n, \quad (39)$$

from which we can isolate  $u_l^{n+1}$  to get our update rule.

For clarity, the operator  $\delta_{\Delta U} \delta_{\Delta U}$  is evaluated as

$$\delta_{\Delta U} \delta_{\Delta U} u_l^n = \frac{1}{h^2} \left( \sum_{u \in U_l} \delta_{\Delta U} u - |U_l| \delta_{\Delta U} u_l^n \right). \quad (40)$$

One can also extend models by adding dampening. For example we can build 1D wave equation based model with dampening by starting with the equation

$$u_{tt} = c^2 u_{xx} - 2\sigma_0 u_t + 2\sigma_1 u_{txx}, \quad (41)$$

where  $\sigma_0$  is a constant controlling frequency independent loss and  $\sigma_1$  controlling frequency dependent loss [2], and discretizing it like

$$\delta_{tt}u_l^n = c^2 \delta_{\Delta U} u_l^n - \sigma_0 \delta_t u_l^n + 2\sigma_1 \delta_t \delta_{\Delta U} u_l^n. \quad (42)$$

## 10. IMPLEMENTATION

A reference implementation in MATLAB is available online<sup>2</sup>, providing a class for building models by creating strings, connecting them, and adding boundary conditions, after which one can compute the  $\delta_{\Delta U}$  and  $\delta_{\Delta U} \delta_{\Delta U}$  operators.

Using the main class, an implementation of a 1D wave equation based model is provided, with and without dampening. An implementation of the linear bar model is also provided, again with and without dampening, showcasing the use of the  $\delta_{\Delta U} \delta_{\Delta U}$  operator.

The repository also contains a work in progress — though usable — interactive GUI application written in C++ / JUCE, which will serve as a test bed for various model topologies, extensions and exiters.

## 11. FUTURE WORK

The most pressing issue for the practical utility of the current research is to show stability conditions for a given graph. This could likely be done using the energy method. Tree-like graphs seem to have excellent stability conditions

comparable to the 1D wave equation, which is stable whenever  $ck/h \leq 1$  [1, Chapter 6]. However, since we can build meshes equal to the 2D wave equation, there must also be a case where the stability condition is  $ck/h \leq 1/\sqrt{2}$  [1, Chapter 11].

Some feature of the topology of a model must be the determining factor for the stability condition. Finding a condition such that the stability of a given graph can be known before evaluating it is of vital importance if algorithms such as these should ever be used by non-experts.

Like other finite difference models, we need a way of exciting the system. Many choices are available ranging from simple initial conditions, to advanced bow, hammer or reed excitation (see e.g. [17]). Any exciter applicable to the 1D wave equation should be applicable to graph based models.

More work can be done investigating the frequency content of graphs. For example, how does the relationship between the lengths of the string in the branching topology affect the modal frequencies? and what happens when we introduce cycles into our model? How do models behave when built using stiff strings or bar models?

Throughout this paper we have considered junctions between strings of equal stiffness. One could derive rules similar to the ones in this paper, but for strings with differing stiffness, which would lead to even more ways of building graphs.

Lastly, there is of course a lot of time to be spent exploring the various timbres and artistic uses of graph based physical models, and related to that, new interfaces for controlling and performing with such models.

## 12. CONCLUSION

In this paper we have explored some of the fundamental concepts of constructing and analysing graph based physical models for sound synthesis.

Starting with a review of the 1D wave equation and one of its derivations using mass-spring networks, we showed how to build a second order difference operator applicable to the end of a string which branches out into  $N$  other strings. Using this we are able to construct any kind of cyclic and acyclic graph.

When a model is built without string segments longer than one, we call it dense. We showed that using our new difference operator, we can build a model which is equivalent to a discretization of the 2D wave equation. We then created a grid using hexagons and found that it was superior to the rectangular grid with regards to stability and computational complexity.

Like other finite-difference schemes we needed to decide on some boundary conditions for the edge nodes in our models. We reviewed the Neumann and Dirichlet boundary conditions and showed how to implement them for graph models.

By reasoning about the number of pendant nodes in a graph, we demonstrated that acyclic models have a computational complexity of  $\mathcal{O}(N)$ , and that cyclic graphs have a worst case complexity of  $\mathcal{O}(N^2)$ .

Using common methods for analyzing vibrating systems, we showed how to use the parameters of our model to set

<sup>2</sup> <https://github.com/PelleJuul/graph-physical-models>

up a linear system in canonical eigenvalue problem form, from which we can compute the modal frequencies and shapes.

Some extensions to our models were examined, including how to apply our results to a linear bar models and how to add dampening to a system.

Lastly we discussed topics for future research including a call for a more rigid mathematical analysis of the models, experiments with various excitation mechanisms, investigations of non-homogeneous models, and a wish for future artistic and interaction related endeavors related to graph based physical modeling.

### Acknowledgments

The authors would like to express sincere gratitude to Silvin Willemsen and Nikolaj Andersson for their feedback, ideas and enthusiasm for this project, as well as a general thanks to all of the staff and students at the SMC master's programme at Aalborg University Copenhagen, for building a great environment for learning and experimentation. This work is partially supported by NordForsk's Nordic University Hub Nordic Sound and Music Computing Network NordicSMC, project number 86892.

### 13. REFERENCES

- [1] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. John Wiley and Sons, 2009.
- [2] ———, "A modular percussion synthesis environment," in *Proc. Int. Conf. Digital Audio Effects (DAFx-09)*, Como, Italy, 2009.
- [3] A. L. Claude Cadoz and J. L. Florens, "Cordis-anima: A modeling and simulation system for sound and image synthesis: The general formalism," in *Computer Music Journal Vol. 17, No. 1*, 1993, pp. 19–29.
- [4] G. Eckel, F. Iovino, and R. Causs, "Sound synthesis by physical modelling with modalys," in *Proceedings of the International Symposium of Music Acoustics*, 1995.
- [5] D. Murphy, A. Kelloniemi, J. Mullen, and S. Shelley, "Acoustic modeling using the digital waveguide mesh," 2007.
- [6] D. Rocchesso and F. Fontana, "The sounding object," 2003.
- [7] J. C. Strikwerda, *Finite Difference Schemes and Partial Differential Equations, Second Edition*. Society for Industrial and Applied Mathematics, 2004.
- [8] R. R. Rosales, "Force-directed drawing algorithms," 2001.
- [9] S. Bilbao, B. Hamilton, R. Harrison, and A. Torin, "Finite-difference schemes in musical acoustics: A tutorial," in *Springer Handbook of Musical Acoustics*, 2018, pp. 249–384.
- [10] B. Hamilton and S. Bilbao, "Hexagonal vs. rectilinear grids for explicit finite difference schemes for the two-dimensional wave equation," in *21st International Congress on Acoustics*, 2013.
- [11] J. O. Smith III, "On the equivalence of the digital waveguide and finite difference time domain schemes," 2004.
- [12] M. Karjalainen, P. Huang, and J. O. S. III, "Digital waveguide networks for room response modeling and synthesis," in *Proc. of the 118th AES conference*, 2005.
- [13] J. O. Smith III, "Aspects of digital waveguide networks for acoustic modeling applications," 1997.
- [14] G. Essel and P. R. Cook, "Banded waveguides on circular topologies and of beating modes: Tibetan singing bowls and glass harmonicas," 2002.
- [15] D. T. Murphy, "Digital waveguide mesh topologies in room acoustics modelling," 2001.
- [16] S. G. Kobourov, "Force-directed drawing algorithms," 2004.
- [17] F. Avanzini, M. Rath, D. Rocchesso, and L. Ottaviani, "The sounding object," D. Rocchesso and e. Federico Fontana, Eds., 2003, ch. 8.

# ADEPT: Exploring the Design, Pedagogy, and Analysis of a Mixed Reality Application for Piano Training

Lynda Joy Gerry, Sofia Dahl, Stefania Serafin

Aalborg University, Copenhagen  
Department of Architecture, Design and Media Technology  
lyn, sof, sts@create.aau.dk

## ABSTRACT

One of the biggest challenges in learning how to play a musical instrument is learning how to move one's body with a nuanced physicality. Technology can expand available forms of physical interactions to help cue specific movements and postures. This cueing can reinforce new sensorimotor couplings to enhance motor learning and performance. Using Mixed Reality (MR), we present a system that allows students to share a first-person audiovisual perspective with a piano teacher. Students place their hands into the virtual gloves of a teacher. Motor learning and audio-motor associations are reinforced through motion feedback and spatialized audio. The Augmented Design to Embody a Piano Teacher (ADEPT) application is an early design prototype of this piano training system.

## 1. INTRODUCTION

This paper presents the Augmented Design to Embody a Piano Teacher (ADEPT) system and explains the motivation for its design to train piano playing. The ADEPT system is a Mixed Reality (MR) application in which students share a first-person, embodied perspective with a piano teacher to facilitate learning the proper finger, hand, wrist, and torso configurations to produce various sounds on the piano. The ADEPT system virtually overlays a video recording showing the teacher's hands on top of the student's own hands into the students head-mounted headset. The ADEPT system is inspired by embodied music cognition, which emphasizes the role of human bodily movement in music perception and performance, and makes muscle memory the main focus of musical training and analysis [1]. This differentiates the ADEPT system from the prevailing approaches which often analyses skill of playing in terms of key press onset and release [2, 3]. Instead, embodied music cognition views piano playing as a nuanced and specialized bodily knowledge [4]. Previous technology-enhanced piano training applications aim to train playing the correct key(s) versus training optimal sound-producing movements [5]. Rather than memorizing each individual note to be played, trained musicians use

muscle memory and fine motor skills. Consequently, the ADEPT system is designed to reinforce muscle memory rather than rote learning of symbolic musical notation, using visual and audio perspective-taking as a tool to guide sensorimotor skills development, combined with motion tracking and feedback to enhance musical action cueing.

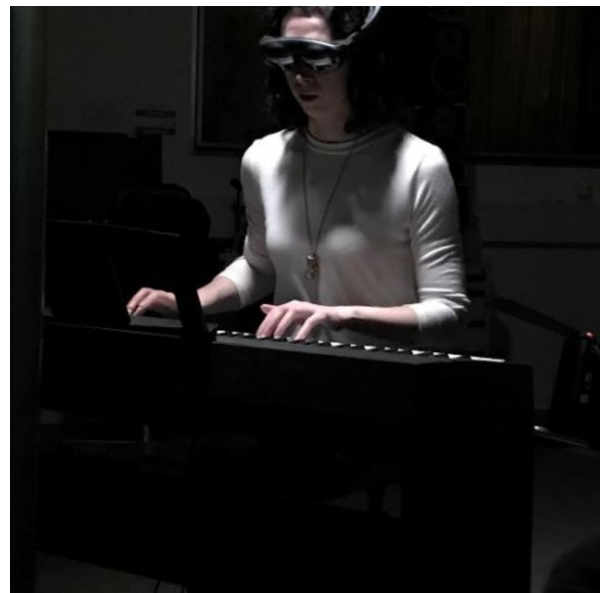


Figure 1. Image showing a user performing piano through the ADEPT system on the Magic Leap headset.

## 2. RELATED RESEARCH

The pedagogy of the ADEPT system is based on training bodily knowledge and sensorimotor skills. The idea of using virtual overlays to create the illusion of sharing an embodied perspective with the piano teacher is inspired by the instructional technique of having a piano student place their hands on top of the piano teachers hands while playing scales or simple tunes [4]. Moreover, the ADEPT system aims to train muscle memory for novice piano students, specifically in knowing how to move to produce certain sounds on the piano. ADEPT is geared towards refining the students' experience of their own body and movements towards developing a more nuanced bodily knowledge more akin to that of an expert pianist.

The ADEPT system is inspired by the [?, ?, ?] frame-

Copyright: © 2019 Lynda Joy Gerry, Sofia Dahl, Stefania Serafin et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



work proposed by Xiao and Ishii [6]. This framework emphasizes musical expression and takes an experiential perspective towards developing musical expertise. The target student users for ADEPT are adult musicians new to the piano. Thus, ADEPT trains piano playing in a very imitation-based way, hoping that this can ultimately help the students to internalize the muscle patterns while playing the piano.

## 2.1 Music and Piano Trainer Applications

Piano training applications are a subset of technology-enhanced musical education systems [7]. The goal of these systems is to enhance the learning environment for greater individualization, real-time feedback, and multi-sensory cues. Virtual content augmentations gamify the experience of learning, and increase motivation and interest in learning the piano [8]. In general, previous piano training applications have involved three primary components: using visual cues to indicate which key(s) to press, presenting alternative visualizations of musical notation, and increasing sight-reading proficiency.

The primary augmentation for piano training has been to present visual cues on top of piano keys to guide keyboard playing. The training emphasis of these systems is learning to play the correct keys according to the musical score. For example, key press has been indicated through line pointers above the key highlighting the next key to be pressed [8] and a red highlight with incorrect key press [9]. Another variation of training correct keyboard fingering is to show how long a key should be pressed. For instance, HoloKeys presents a green glow on the key at the moment it should be pressed, which disappears as soon as it should be released [10]. Similarly, the P.I.A.N.O. system involves highlighting the current and next key to play, and uses a Guitar Hero approach with dropping lines approaching the keyboard from a far distance to demonstrate which notes to play, and for how long they should be played [9]. In addition to presenting visual cues, some music trainer applications also present auditory cues to piano students using real-time sonification to analyze the student's playing sounds and provide auditory feedback when the student plays the wrong note [11].

Visual cues also help to train students' abilities to imagine some of the sound-movement feedback cycles vital for playing a musical instrument. Specifically, expressive parameters might be difficult for a music teacher to communicate to a student. Hence, for example, the PianoFORTE system helps to train expressive features of piano playing by providing visualizations for dynamics, tempo, articulation, and hand synchronization [12]. Similarly, the Andante system utilizes visual animations of people of various weights walking across the keyboard in sync with the keys as they are played in sequence [13]. This is designed to help the student to visualize another aspect of correct keyboard fingering, which is knowing how hard to press the keys.

It is important to note that the goal of many technology-enhanced piano systems is to compliment and to function as an add-on to traditional piano teaching in-person with

a piano teacher. For example, the Piano Tutor [14] is intended to help students practice in between lessons with the teacher, and we intend for ADEPT to function similarly. Additionally, however, technology-enhanced musical education has the possibility to make the learning process easier and more intuitive, as well as to invite new avenues for personal reflection on one's own performance and process. Specifically, one of the major benefits of these systems is that they can help the student to develop online self-analysis skills while playing [14].

### 2.1.1 Passive Haptic Learning

Another approach of augmentation is to provide haptic information. For example, the MobileMusicTouch is a piano training technique haptic feedback with vibration motors inserted at the metacarpophalangeal joints (knuckles) of a glove to help the student to understand which fingers to use to play which keys [15]. One advantage of Passive Haptic Learning (PHL) [16] is that it allows the student to memorize fingering patterns for playing various short music pieces without requiring conscious effort or attention. Indeed, participants were able to retain fingering patterns while wearing MobileMusicTouch even when viewing a film and playing a memory game [15].

Although passively learning finger patterns while attending other stimuli can be convenient, it raises the question as to whether it is the right approach to teach the overall movement control needed for music performance. As noted by Xiao [6], most of the technology-enhanced musical education systems have a focus on the score and the associated errors.

## 2.2 Mixed Reality Applications

Mixed Reality (MR) environments present virtual overlays and augments that directly interact with the users physical environment, real world objects, and natural movements [17]. Augmented Reality (AR) is a subset of MR, and the two terms are often used interchangeably. The emphasis on MR in this case is to highlight the main interaction with real-world objects (i.e., playing a real piano), which is supported by virtual augments, rather than having the primary interaction be with virtual content within a real environment (i.e., flying a virtual plane that follows the constraints of the real physical environment), as is often the case with popular AR applications. Previous educational benefits have been demonstrated in MR based on its ability to extend embodied actions with high-fidelity multisensory stimuli and real-time feedback [18], specifically by using various types of cueing for different bodily actions.

Previous piano training MR systems that help cultivate higher levels of musicianship focus on facilitating playing a system from memory, and training skills related to musical improvisation. For example, Handel [19] presents a visual overlay of sheet music notation on top of the pianist's fingers while they attempt to play the piece from memory. Similarly, systems like Stanford's Pianolens facilitates learning and rehearsing new music with an interactive sheet music display that imitates a piano roll [20].

More recently, a few MR piano training applications have projected fingers and hands on top of the keyboard or on top of the player's hands to guide piano playing [21]. These systems use various 2D and 3D graphical representation of an experienced pianist's hands and fingers. For example, Teomirn uses the HoloLens display and projects a geometric 3D display of a virtual hand that the student can place their own hand into to follow along while playing. However, the virtual hand is not very realistic, and only roughly helps to guide the student's movements. The ADEPT system instead depicts precise finger, wrist, and upper arm positions, angles, rotations, and movements of the piano teacher for the student to follow. Moreover, these MR designs are mostly focused on helping the student to know which finger to use to play which key.

As noted by Xiao [6], the primary element that is lacking in all of these systems is an emphasis on the bodily movements of the piano student, and training how to move the body in the correct ways as a focus for the training and technology-enhanced design. MirrorFuge is one exception [21]. The MirrorFuge system presents a projection-mapped video stream of a pianist's hands on top of a physical keyboard. Subjects reported that seeing the hands of an instructor was more helpful than screen-based instruction or abstract visual cues (a small dot indicated keys pressed by the expert pianist) [21]. Due to a small sample (5 subjects), the results were not statistically significant but suggest that using the first-person perspective to present the instructor's hands from the same egocentric orientation decreased the amount of time that it took for students to learn simple melodies. These results are promising for the development of AR systems focusing on sharing an embodied perspective with an expert pianist who guides the student's movements, and this is precisely the target of the ADEPT system. In the next section, we describe virtual embodiment and introduce the concept of augmented embodiment, which is a key design principle behind the ADEPT system.

### 2.3 Virtual and Augmented Embodiment

Virtual embodiment is a technique used in virtual reality (VR) to create the illusion of becoming a virtual avatar [22]. Virtual embodiment allows users to see and hear from the first-person, embodied perspective of another real person or a virtual character and perform a task together, such as hand-drumming [23, 24]. Synchronous stimuli presented to the visual system in VR, combined with the physical body in reality, create various bodily illusions that make the user feel that the avatar body is their own body [25]. This induces strong psychological effects on the user, specifically identification with the avatar body [26]. Taking the perspective of an expert in virtual embodiment studies has been shown to increase confidence and improve performance on related tasks [4, 27–30].

In addition to the strong psychological and learning effects of virtual embodiment, learning a new task from a first-person perspective improves retention of instructional material. For example, sequences of chess movements were more accurately retained when presented in

first-person perspective in VR, as compared to exocentric, screen-based perspective [31]. Memory retention is stronger when events are presented from an egocentric versus an exocentric or allocentric point of view [32]. To this end, egocentric VR has been used in memory rehabilitation to increase procedural learning in patients with memory impairments, transferable to real-world environments [33]. Thus, delivering piano instruction in the ADEPT system should support better retention of the finger sequences involved in playing and better performance outcomes in a shorter time.

Another reason why virtual embodiment may increase fine motor skills is that the observation of hand movements elicits motor-evoked potentials in the observer in the specific muscles that would be involved in executing the movements [34]. Research on the mirror neuron system in humans indicates that during the observation of another person's bodily state, the same neural structures are activated in the observer [35]. This effect is even stronger when observing hand movements from a first-person perspective, when one's own hand positions and movements are congruent with those observed [32]. Moreover, visual feedback using video is very common for both piano teachers and students to adjust and adopt better postures for playing, and this has been enhanced with 3D visualization of postural information [36].

We here introduce the concept of Augmented Embodiment, in which the user's point of view is not fully overtaken by that of another virtual avatar or real person, but is instead augmented with a virtual projection of another person's embodied perspective super-imposed on top of one's own. This is the core design feature implemented in the ADEPT system. Augmented embodiment can allow a student to perform and observe an action at the same time, from the same view-point, and in the same way as a teacher with real-time feedback, a phenomenon not possible in physical reality [37].

### 2.4 Embodied Music Cognition

Adaptive and immersive virtual environments involve new strategies for sensorimotor training and can induce brain reorganization, presently tested therapeutically in clinical populations recovering from stroke [38]. In one such study, a Virtual Piano Trainer system found that adaptive motion feedback providing information about position adjustments in the fingers and hands increased the accuracy and duration of muscle activity, expediting the recovery of these fine motor capabilities. Motion feedback support has previously been shown to increase skills ability and retention while learning technical motor skills [39]. Multisensory feedback can encourage plasticity within the sensory-motor cortex and enhance motor performance [40].

## 3. SYSTEM OVERVIEW AND DEVELOPMENT

The ADEPT system is programmed in Unity version 2018.1.9f2-MLTP8.1 with C# and uses the Magic Leap Package Manager with Lumin SDK v0.19.0 and Device Driver version 0.94. The setup involves virtually overlay-

ing a video recording from a head-mounted camera worn by the teacher into the student's head-mounted display (HMD), in our case, the Magic Leap 1 (ML-1, see Figure 2). The Unity programming environment used the Media Player example from the Magic Leap Unity SDK and played the video on top of a spherical mesh. The system displays the teacher's hands on top of the student's own hands, and highlights each key when pressed. We used Real-World Reconstruction in Unity, which uses ML-1 depth sensors and spatial computing to track the environment in real-time, which helped for tracking the location of the user's physical piano. AR Registration of the physical piano was accomplished using Vuforia Engine version 8.1 and four fiduciary markers at the four corners of the piano. The cameras on the ML-1 register the four corners of the piano and Vuforia image recognition from piano models registers the approximate positions of the keys.

The system is a work in progress, and this section reviews the implementation that has been developed so far as well as features that are continuing to undergo further development. For the development we recorded a young piano teacher with 15 years of experience who had taught piano for 3 years.

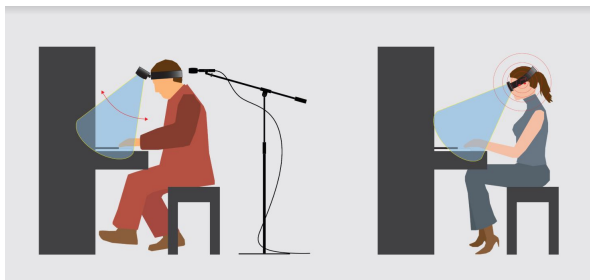


Figure 2. This is the overview of the system. The left side of the figure shows the piano teacher wearing the head-mounted camera with the recording microphone directly behind his head. On the right, we see the user (piano student) wearing the Magic Leap headset with spatialized audio. The teacher and the student have the same audiovisual perspective and orientation on the piano.

### 3.1 User Experience

Participants were seated at a Yamaha P45 digital piano with controlled lighting for optimal display of the virtual overlays, and ability to still see one's own piano and hands clearly. In the videos, the piano teacher is seated at an upright grand piano. Students were instructed to listen to the teacher's instructions, and then to place their hands on top of hers and play along with her when she instructed them to do so. The video sequences involved first an observation sequence in which the teacher showed the fingering movements. Then the teacher would instruct the students to get ready and place their fingers, and she would count down from three for when they should start playing along with her.

### 3.2 Visual Environment

The visual environment consists of 360 degree video footage that has been captured using the Garmin VIRB 360 camera with a head-mounted strap (made by Go Pro) to create a head-mounted camera worn by the piano teacher (see left panel in Figure 2). Multiple recordings were done of different short sequences in which the piano teacher explains how the fingers are numbered, how to place the hands on the keyboard, and how to play basic scales.

One concern with using overlays is that the visual environment quickly becomes cluttered. Even with the virtual piano and piano keys spatially aligned with the user's physical piano (using Vuforia ARCamera and TargetMarkers), early prototype testing indicated visual clutter. By creating an alpha channel in Adobe After Effects, we were able to reduce most of the visual noise from the piano by only having the virtual hands and the current key note pressed in the video with a blue shader to highlight it.

The first recordings were done with the top of the grand piano removed with the intention of allowing the student to see the hammer-head moving with the key depressed. When superimposed in the MR headset and environment, the visual environment appeared very busy and cluttered. Thus, inspired by the MirrorFugue project [21], we replaced the top of the grand piano and recorded the 360 video footage such that the top of the piano created a reflection of the piano teacher (see Figure 3). There are three reasons that it is valuable to show the virtual reflection of the piano teacher. The first is to cultivate greater social presence [41], so that the student can see the face of the piano teacher while she speaks and gives verbal instructions. Social presence is the feeling of being there together with another real person in an online, digital, or virtual remote collaboration, and it has shown to have significant effects on user satisfaction [41]. The second reason is modeled from the design of virtual embodiment studies in VR. The setup for these studies involves a virtual mirror, in which the user can see him or herself reflected as the avatar they embody [22]. Thus, we wanted to include a similar 'virtual mirror' to create a greater sense of psychological identification with the piano teacher that might improve confidence and performance. The final reason for including the virtual reflection is that eventually we would like to visually annotate the virtual reflection of the teacher with real-time motion feedback from the student's movements to help the student notice the difference between their movements and those of the teacher. A related example comes from the i-Maestro musical training application for violin [42], which used a 3D Augmented Mirror showing synchronized video and motion data for bowing trajectories on the violin. This 3D Augmented Mirror enhanced students' understanding of the correct bowing techniques and body postures for playing the violin.

### 3.3 Audio Recording and Feedback

The teacher was seated at a Yamaha upright piano situated in a large room (volume: around  $400\text{ m}^3$ ) against a wall. A Sennheiser Ambeo VR Mic was positioned upright slightly above and behind the expert's head in order



Figure 3. This figure is a model representing what the user sees inside the headset in the Mixed Reality environment. The user sees the virtual hands of the teacher on top of the physical keyboard. There is a blue highlight on the currently pressed key. The user also sees the face and upper body of the pianist in the reflection on the upright grand piano.

to get a binauralised sound source. All sound files were encoded to B-format thanks to Ambeo A-B Format Converter then decoded to binaural format through FB360 Converter.

### 3.3.1 Audio Spatialization

Spatialized audio is used in ADEPT to differentiate the sounds of the teachers playing from the students playing, ultimately so that the student can learn to attenuate to their errors in pitch and timing while playing. Different development versions of the system have explored various ways to spatialize audio. The first design was to spatially misalign the teacher's point of view from the actual physical piano of the student. Here, we offset the sound virtually by using the Facebook 360 Audio Spatializer plugin in Reaper. The sound was made to spawn at locations directly above and to the left of the user's physical piano, above and to the right, matching the user's piano, and also slightly below. This spatial offset design was inspired by the visual offset design used by Xiao Xiao in MirrorFugue [21]. Preliminary data from pilot testing has shown that students find the audio spatialization offsets of slightly above and to the right and left to be the most comfortable and intuitive to follow than audio from below or matched to their piano location.

The second audio spatialization technique we are exploring is using binaural recordings rather than ambisonics to capture the spatial perspective of the teacher. Based on the previous design results, we will be offsetting the spatialized spawning location of this binaural audio to be slightly above the user's piano. In future user studies, the binau-

ral audio will also be accented by personalized 3D sound that account for the shape of the user's ears, which may enhance the effect.

## 4. FUTURE DIRECTIONS

Rather than using 360 degree video, we are currently exploring using volumetric video capture of the teacher's hands using the Structure Sensor and the Microsoft Kinect 2 depth-sensing cameras. We will also explore using Leap Motion hands to display the teacher's hands with a custom mesh of the teacher's actual hands from photogrammetry scans. A benefit for both of these designs is the possibility to easily resize and rescale the 3D hand models to more appropriately fit the student. Between these two designs, we will select the one that seems to have the most optimal display quality for the project. The virtual mirror will be present, not as now with the real reflection of the teacher in the piano, but instead as a darkened video screen that we will position in the Unity environment to spawn in the reflection of the piano. The goal for the design moving forward is to present the minimum amount of information necessary to facilitate user performance with the highest degree of quality. Currently, we are also developing a display with volumetric video capture and 3D motion capture data overlaid, which we could project as a 3D augmented Mirror like the i-Maestro project [42]. We hope that this can help the student to understand motion trajectories for the piano, combined with real-time motion feedback visual annotations.

Currently, the visual component of our design prototype has only explored adjusting the opacity of the video overlay of the teacher's visual perspective. In future user testing, we are also exploring having the teacher's body and hands appear next to the student, or above the student's hands on top of the keyboard, to explore if this makes it easier for students to follow the fingering patterns of the teacher. We will also test having the teacher's hands to be present from a first-person perspective aligning with the student's visual perspective in the observation phase, and then to appear above the student's hands during the "play along" phase. And finally, another visual design prototype we plan to test involves having the teacher seated next to the student, and to allow the student to slide over on their piano bench to "sit into" the body and first-person perspective of the teacher.

We are also implementing a user interaction for the student to trigger hearing more or less from teacher's point of view. This means that the user will be able to select how much she sees or hears from the teachers point of view. That is, in future user testing, the system will be presented as though the teachers embodied sense reality is something that the student can choose to enter into. For instance, the visual overlay will be first set around 50 percent opacity, but the student can decrease or increase the opacity to see more or less from their own or the teacher's point of view. Similarly, the student can lean her head forward into a virtual sphere to hear more strongly from the teachers perspective.

Audio signal processing of the piano playing is being in-



corporated using pitch Pure Data, so that we can have an auditory analysis using the frequency to MIDI converter, pitch detection, and tempo tracking. We will use information to detect errors in playing the incorrect key or the correct key with the wrong finger. Ultimately, this analysis will allow the system to provide real-time feedback with visual motion annotation, pitch slider effects, and potentially also haptic feedback to the student.

The role of haptic feedback in learning musical movements has becoming increasingly vital to this project, and we are exploring the possibility of using a wrist actuator with five vibrating motors to represent movements for each of the fingers. Ideally, this haptic wearable device should be non-intrusive for the student's playing, which is why we are planning to test a wrist-worn device, as opposed to previous gloves which were not worn while playing the piano.

Lastly, we are collaborating with piano teachers at the Rhythmic Music Conservatory in Copenhagen to collect qualitative data on piano pedagogy towards a participatory design approach to make the technology more specific to enhance and compliment the students' learning experience. Additionally, we continue to conduct user testing with the system to address ongoing challenges during development. One of the goals in doing this is to better understand music and piano pedagogy and to explore the ways that the system can actually compare to and enhance traditional face-to-face piano instruction with a teacher. We hope that the system can eventually deliver piano instructions in a way that acknowledges the deep and complex history of piano pedagogy techniques, whereas the current focus in design and development has been much more focused on previous technology-enhanced systems for training to piano.

#### 4.1 Motion Capture and Feedback

We are beginning to explore the major features of musical movement that distinguish expert pianists (masters) from more novice pianists, and also to explore the movement patterns characteristic of very novice adult students learning the piano for the first time. The purpose of this motion analysis is to create an evaluation metric for performances as an outcome to target as a result of the training.

Motion capture from the teacher will be captured using Leap Motion (mounted above to approximate head position) and photo-electric sensors with an optitrack system. Motion data from the student will be captured using the Leap Motion (mounted above at same coordinate ratio to where it had been mounted for the teacher) and Microsoft Kinect as both infrared and skeletal tracking systems. Motion data will capture finger and joint positions from both hands, as well as temporal sequencing and timing of movements.

Novice pianists focus on the extremities while playing, particularly having the correct fingering patterns, whereas expert pianists feel the music through their entire body [43]. Arms, wrists, and upper torso posture movements are usually introduced and trained at more advanced stages of learning, and are trained in isolation [43]. Learning how to move the body relies heavily on imitation. Observation and imitation of expert performance allows students to experi-

ence how music is felt in the body of another player [6]. Moreover, visual feedback about motor performance, accuracy, and adjustments can help improve reflection on one's own body and performance, and is often used in piano performance [36]. A previous training system used electromyography (EMG) to measure muscle activity in the thumb and successfully delivered biofeedback to help students achieve optimal muscle activation. While motion feedback does not deliver information directly about muscular activations, this still indicates a strong potential for motion feedback to improve motor performance while playing piano [44]. Thus, the primary goal of the motion feedback is to support successive adaptations in motor learning and performance. In order to monitor and evaluate the performance, the movements of the students will be captured with similar means.

#### 4.2 Potential Challenges

Previous AR piano training applications have used Vuforia object tracking for matching the virtual and real pianos, but the distance at which the virtual piano appears from the user is still not quite correct. Thus, a potential challenge for the next stage of development is to ensure that the 3D model of the piano teacher's hands appears at the current depth and distance from the user. We found that we were able to control this apparent distance with the 360 degree video, but depth perception accuracy in the Magic Leap headset is not as precise as it could be. Specifically, we will want to make sure that the spatial configurations of the hands are easy to see and understand in a three-dimensional way, and that the user can perceive the distance and depth of the fingers accurately. A related challenge is that the teacher's hand blends in with the white keys on the keyboard, so we might need to put an outline around the hand and fingers, shade the hand with a color shader, or have the teacher wear colored gloves to increase the contrast.

Visual and aural latency could be potential challenges that could disrupt the user experience, specifically if the two sensory channels are out of sync. In future developments, the auditory mix between teacher's and student's sound could be a bit problematic if there is latency between visual and auditory information.

#### 4.3 User Studies

Future user studies will be conducted on the binaural audio perspective-taking to see which settings optimize user experience. We will also explore adding user interactions to trigger the intensity of the perspective taking, and measure which decisions users choose to make at which time points to gain a sense of the usefulness of the user interactions, and which user actions should be programmed to trigger those visual and auditory effects.

#### 4.4 Experimentation

We will compare two groups of students learning either with the ADEPT system or with the same content presented on a 2D video screen. The main target of experi-

mentation is to explore how well students can actually follow the teacher. Thus, we will be measuring interpersonal synchrony comparing motion capture data recorded from the teacher with that of the student. Our hypothesis is that interpersonal synchrony during training will predict better performance out of the system. Afterwards, we will ask the students to play the same sequences from memory and measure performance accuracy using motion capture analysis with the Musical Gestures toolbox in Matlab and video analysis using Elan Software for Transcription. We will also have an expert panel of professional piano instructors who will rate the performance of the users who had been trained with the system, as compared to just watching a video.

## 5. DISCUSSION AND CONCLUSIONS

In this paper we presented ADEPT, a system for facilitating learning to playing the piano. The ADEPT system aims at teaching musical movements on the piano in an embodied way so that the student learns to move like a professional pianists. We use audiovisual perspective taking with a piano teacher to help students orient to the correct hand, finger, wrist and upper torso positions for sound-producing movements on the piano. By introducing the notion of Augmented Embodiment, the student can see and hear a blend of his or her own body and that of the teacher from a first-person perspective. Increased user interactions to control the intensity of the audiovisual perspective taking are currently being implemented and tested. Initial prototype testing indicates that positioning the sounds of the teacher's piano playing slightly above the student optimizes comfort, and further testing will determine if this also optimizes performance. In conclusion, the ADEPT system offers a new design technique using modern Augmented Reality technology with audio-visual perspective taking and feedback to help teach piano to novice students.

## Acknowledgements and Author contributions

We would like to thank Christie Anne Jeannine Laurent for her patience and excellent piano playing and instruction to create the audiovisual instructional content. We would also like to thank Daniel Ludwig for creating the figures that demonstrate the overall design schema for the project and the anonymous reviewers for their helpful comments in improving the paper.

Author Gerry was responsible for designing and developing the application, as well as collecting initial prototype data. Author Dahl designed data collection and analysis tools for motion capture and analysis. Author Serafin designed data collection and analysis tools for audio spatialization. All authors participated in writing and revising the manuscript.

This work is supported by NordForsk's Nordic University Hub Nordic Sound and Music Computing Network NordicSMC, project number 86892.

## 6. REFERENCES

- [1] M. Leman, *Embodied music cognition and mediation technology*. Mit Press, 2008.
- [2] S. Shirmohammadi, A. Khanafar, and G. Comeau, "MIDIATOR: A tool for analyzing students piano performance," *Revue de recherche en ducation musicale*, vol. 24, pp. 35–48, 2006.
- [3] S.-H. Lee, "Using the personal computer to analyze piano performance," *Psychomusicology: A Journal of Research in Music Cognition*, vol. 8, no. 2, p. 143, 1989.
- [4] R. Lindgren, "Generating a learning stance through perspective-taking in a virtual environment," *Computers in Human Behavior*, vol. 28, no. 4, pp. 1130–1139, 2012-07. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0747563212000234>
- [5] X. Xiao, "Reflecting music through movement: a body-syntonic approach to playing [with] the piano." 2016. [Online]. Available: <https://dspace.mit.edu/bitstream/handle/1721.1/107576/974646581-MIT.pdf?sequence=1>
- [6] X. Xiao and H. Ishii, "Inspect, embody, invent: A design framework for music learning and beyond," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, 2016, pp. 5397–5408. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2858036.2858577>
- [7] G. Percival, Y. Wang, and G. Tzanetakis, "Effective use of multimedia for computer-assisted musical instrument tutoring," in *Proceedings of the international workshop on Educational multimedia and multimedia education - Emme '07*. ACM Press, 2007, p. 67. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1290144.1290156>
- [8] J. Chow, H. Feng, R. Amor, and B. C. Wunsche, "Music education using augmented reality with a head mounted display," *User Interfaces*, vol. 139, p. 8, 2013.
- [9] M. Weing, A. Rhlig, K. Rogers, J. Gugenheimer, F. Schaub, B. Knings, E. Rukzio, and M. Weber, "P.i.a.n.o.: enhancing instrument learning via interactive projected augmentation," in *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication - UbiComp '13 Adjunct*. ACM Press, 2013, pp. 75–78. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2494091.2494113>
- [10] D. Hackl and C. Anthes, "HoloKeys - an augmented reality application for learning the piano," in *Proceedings of the 10th Forum Media Technology and 3rd All Around Audio Symposium*, 2017-11, pp. 140–144.
- [11] S. Ferguson, "Learning musical instrument skills through interactive sonification," in *Proceedings of the 2006 conference on New interfaces for musical expression*. IRCAMCentre Pompidou, 2006, pp. 384–389.

- [12] S. W. Smoliar, J. A. Waterworth, and P. R. Kellock, "pianoforte: A system for piano education beyond notation literacy," in *ACM Multimedia*, vol. 95, 1995.
- [13] X. Xiao, B. Tome, and H. Ishii, "Andante: Walking figures on the piano keyboard to visualize musical motion," in *New Interfaces for Musical Expression (NIME)*, 2014, pp. 629–632.
- [14] R. B. Dannenberg, M. Sanchez, A. Joseph, P. Capell, R. Joseph, and R. Saul, "A computer-based multimedia tutor for beginning piano students," *Journal of New Music Research*, vol. 19, no. 2-3, pp. 155–173, 1990.
- [15] D. Kohlsdorf and T. Starner, "Mobile music touch: The effect of primary tasks on passively learning piano sequences," in *International Symposium on Wearable Computers (ISWC) 2010*. IEEE, 2010, pp. 1–8.
- [16] K. Huang, E. Y.-L. Do, and T. Starner, "Pianotouch: A wearable haptic piano instruction system for passive learning of piano skills," in *2008 12th IEEE International Symposium on Wearable Computers*. IEEE, 2008, pp. 41–44.
- [17] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino, "Augmented reality: a class of displays on the reality-virtuality continuum," H. Das, Ed., 1995-12-21, pp. 282–292. [Online]. Available: <http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=981543>
- [18] D. L. Dieker, M. Hynes, C. Stapleton, and D. C. Hughes, "Virtual classrooms: STAR simulator," *New Learning Technology SALT*, vol. 4, pp. 1–22, 2007.
- [19] "Personal contextual awareness through visual focus," vol. 16.
- [20] E. Strasnick, A. Chambers, L. Jiang, and T. Xiaonan, "Pianolens: An augmented reality interface for piano instruction," 2017. [Online]. Available: <https://cs.stanford.edu/people/estrasni/otherprojects/pianolens.html>
- [21] X. Xiao and H. Ishii, "Mirrorfugue: communicating hand gesture in remote piano collaboration," in *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*. ACM, 2011, pp. 13–20.
- [22] B. Spanlang, J.-M. Normand, D. Borland, K. Kilteni, E. Giannopoulos, A. Pomés, M. González-Franco, D. Perez-Marcos, J. Arroyo-Palacios, X. N. Muncunill *et al.*, "How to build an embodiment lab: achieving body representation illusions in virtual reality," *Frontiers in Robotics and AI*, vol. 1, p. 9, 2014.
- [23] K. Kilteni, I. Bergstrom, and M. Slater, "Drumming in immersive virtual reality: The body shapes the way we play," p. 9, 2013.
- [24] D. Banakou, P. D. Hanumanthu, and M. Slater, "Virtual embodiment of white people in a black virtual body leads to a sustained reduction in their implicit racial bias," *Frontiers in Human Neuroscience*, vol. 10, 2016-11-29. [Online]. Available: <http://journal.frontiersin.org/article/10.3389/fnhum.2016.00601/full>
- [25] L. Maister, M. Slater, M. V. Sanchez-Vives, and M. Tsakiris, "Changing bodies changes minds: owning another body affects social cognition," *Trends in cognitive sciences*, vol. 19, no. 1, pp. 6–12, 2015.
- [26] M. Slater and M. V. Sanchez-Vives, "Transcending the self in immersive virtual reality," *Computer*, vol. 47, no. 7, pp. 24–30, 2014.
- [27] M. J. Traxler and M. A. Gernsbacher, "Improving written communication through perspective-taking," *Language and Cognitive Processes*, vol. 8, no. 3, pp. 311–334, 1993.
- [28] J. Guegan, S. Buisine, F. Mantelet, N. Maranzana, and F. Segonds, "Avatar-mediated creativity: When embodying inventors makes engineers more creative," *Computers in Human Behavior*, vol. 61, pp. 165–175, 2016.
- [29] L. J. Gerry, "Paint with me: Stimulating creativity and empathy while painting with a painter in virtual reality," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 4, pp. 1418–1426, 2017.
- [30] D. Banakou, S. Kishore, and M. Slater, "Virtually being einstein results in an improvement in cognitive task performance and a decrease in age bias," *Frontiers in Psychology*, vol. 9, p. 917, 2018.
- [31] M. Slater, V. Linakis, M. Usoh, and R. Kooper, "Immersion, presence, and performance in virtual environments: An experiment with tri-dimensional chess," in *ACM virtual reality software and technology (VRST)*, vol. 163. ACM Press, 1996-07, p. 72.
- [32] R. Watanabe, T. Higuchi, Y. Kikuchi, and M. Taira, "Visuomotor effects of body part movements presented in the first-person perspective on imitative behavior: Movement and first-person perspective," *Human Brain Mapping*, vol. 38, no. 12, pp. 6218–6229, 2017-12. [Online]. Available: <http://doi.wiley.com/10.1002/hbm.23823>
- [33] B. M. Brooks and F. D. Rose, "The use of virtual reality in memory rehabilitation: Current findings and future directions," *NeuroRehabilitation*, vol. 18, no. 2, pp. 147–157.
- [34] L. Fadiga, L. Fogassi, G. Pavesi, and G. Rizzolatti, "Motor facilitation during action observation: a magnetic stimulation study," *Journal of Neurophysiology*, vol. 73, no. 6, pp. 2608–2611, 1995-06. [Online]. Available: <http://www.physiology.org/doi/10.1152/jn.1995.73.6.2608>

- [35] S. D. Muthukumaraswamy, B. W. Johnson, and N. A. McNair, "Mu rhythm modulation during observation of an object-directed grasp," *Cognitive Brain Research*, vol. 19, no. 2, pp. 195–201, 2004-04. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0926641003002994>
- [36] J. Mora, W.-s. Lee, G. Comeau, S. Shirmohammadi, and A. Saddik, "Assisted piano pedagogy through 3d visualization of piano playing," in *2006 IEEE International Workshop on Haptic Audio Visual Environments and their Applications (HAVE 2006)*. IEEE, 2006, pp. 157–160. [Online]. Available: <http://ieeexplore.ieee.org/document/4062532/>
- [37] K. Alaerts, E. Heremans, S. P. Swinnen, and N. Wenderoth, "How are observed actions mapped to the observers motor system? influence of posture and perspective," *Neuropsychologia*, vol. 47, no. 2, pp. 415–422, 2009-01. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0028393208003801>
- [38] S. V. Adamovich, G. G. Fluet, E. Tunik, and A. S. Merians, "Sensorimotor training in virtual reality: A review," *Neurorehabilitation*, vol. 25, no. 1, pp. 29–44, 2009. [Online]. Available: <http://www.medra.org/servlet/aliasResolver?alias=iospress&genre=article&issn=1053-8135&volume=25&issue=1&spage=29&doi=10.3233/NRE-2009-0497>
- [39] S. P. Swinnen, "Information feedback for motor skill learning: A review," *Advances in motor learning and control*, pp. 37–66, 1996.
- [40] H. Sveistrup, "Motor rehabilitation using virtual reality," *Journal of NeuroEngineering and Rehabilitation*, p. 8, 2004.
- [41] S. T. Bulu, "Place presence, social presence, co-presence, and satisfaction in virtual worlds," *Computers & Education*, vol. 58, no. 1, pp. 154–161, 2012.
- [42] K. Ng and P. Nesi, "i-maestro: Technology-enhanced learning and teaching for music." in *NIME*, 2008, pp. 225–228.
- [43] S. Furuya and H. Kinoshita, "Organization of the upper limb movement for piano key-depression differs between expert pianists and novice players," *Experimental Brain Research*, vol. 185, no. 4, pp. 581–593, 2008-03. [Online]. Available: <http://link.springer.com/10.1007/s00221-007-1184-9>
- [44] R. Montes, M. Bedmar, and M. S. Martin, "Emg biofeedback of the abductor pollicis brevis in piano performance," *Biofeedback and self-regulation*, vol. 18, no. 2, pp. 67–77, 1993.



# A Model Comparison for Chord Prediction on the Annotated Beethoven Corpus

**Kristoffer Landsnes**

EPFL

kristoffer.landsnes@epfl.ch

**Robert Lieck**

EPFL

robert.lieck@epfl.ch

**Liana Mehrabyan**

EPFL

liana.mehrabyan@epfl.ch

**Fabian C. Moss**

EPFL

fabian.moss@epfl.ch

**Victor Wiklund**

EPFL

victor.wiklund@epfl.ch

**Martin Rohrmeier**

EPFL

martin.rohrmeier@epfl.ch

## ABSTRACT

This paper models predictive processing of chords using a corpus of Ludwig van Beethoven’s string quartets. A recently published dataset consisting of expert harmonic analyses of all Beethoven string quartets was used to evaluate an  $n$ -gram language model as well as a recurrent neural network (RNN) architecture based on long-short-term memory (LSTM). We compare model performances over different periods of Beethoven’s creative activity and provide a baseline for future research on predictive processing of chords in full Roman numeral representation on this dataset.

## 1. INTRODUCTION

Predictive processing and the formation of expectancies are core capacities of human cognition that also play a fundamental role in music perception and cognition [1–4]. Musical expectancies are essential for processes at different time-scales, such as for musical interaction and synchronization, as well as for musical tension and the play with emotional effects [5, 6]. Musical expectancy has also been understood to be culture- and style-dependent and to be grounded in musical knowledge that is acquired through processes of implicit or statistical learning [1, 7, 8]. The modelling of predictive processing and the formation of expectancies is thus of core importance for computational models of music and requires a learning-based approach.

Musical expectancy has been studied in terms of melody, harmony and rhythm, where the task is to predict the next note, chord, onset or a combination thereof. In the general case of polyphonic music, it is a non-trivial problem to find a consistent representation of musical content and to accurately define what events should be predicted. Many past approaches have, therefore, simplified the problem to predicting a single stream of events from a fixed alphabet, such as melodic notes or chord events. This task is structurally closely related to modelling natural language, and similar approaches have been taken in both fields. Most notably, one can distinguish models that use a finite-length

context, such as  $n$ -gram or  $k$ th-order Markov models, from models that use a latent state to capture longer dependencies, such as hidden Markov models (HMMs) [9] and recurrent neural networks (RNNs) [10, 11].

In this paper, we focus on modeling the prediction of a chord symbol given a harmonic context based on a recent data set comprising expert annotations of the 16 Beethoven string quartets [12], subsumed under nine different opus numbers which formed the basic grouping for all analyses. To this end, we evaluate a standard  $n$ -gram model as well as a state-of-the-art RNN architecture based on long short-term memory (LSTM) [13]. We report and compare accuracy results of the two models over different opera and discuss our results from a technical as well as from a music theoretical point of view.

## 2. METHODS

### 2.1 Data and Preprocessing

The data used for this project contain the expert harmonic analyses of all 16 Beethoven string quartets incorporated in nine opera: Op. 18 (6 quartets), op. 95 (3 quartets) and 7 other opera, each containing one quartet. We group the string quartets by opus number assuming that an opus constitutes a coherent unit of a musical work with pieces that are not independent of each other and should thus be treated as dependent data in the training procedure. Features in the data include global and local keys, beat, time signature, opus and movement numbers. The chord annotation format used in the dataset is a formalised version of Roman numeral notation, the most common music theoretic set of symbols for harmonic analysis. In addition to the key, the scale degree, and the figured bass, the chord annotations include information on suspensions, added notes and pedal notes. Table 1 demonstrates several examples of this annotation format. A more detailed explanation can be found at the official documentation of the data [12]. This annotation format is much richer than what is commonly found in harmonic corpora and thus implies a particularly challenging learning problem.

A total number of 28,095 chord labels are annotated resulting in 1,730 unique items. More than 1,500 chords occur less than 10 times throughout the whole corpus of 16 quartets (908 of which occur only once), while the top 5 chords occur more than 1,000 times throughout all quartets. This distribution is similar to the Zipf distributions

Copyright: © 2019 Kristoffer Landsnes et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Notation	Interpretation
V43	a dominant seventh chord in second inversion
ii%7	a half-diminished seventh chord on the second scale degree
IV6//	a major triad on the fourth scale degree in its first inversion at a phrase end
vi(+9)	a minor triad on the sixth scale degree with an added ninth
V[V	a dominant triad over the pedal tone on fifth scale degree

Table 1. Examples for chord symbols in the dataset and interpretations.

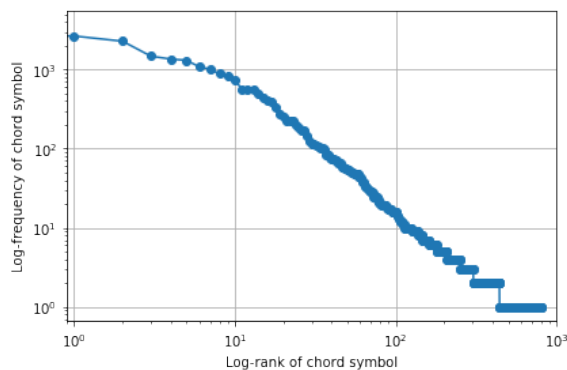


Figure 1. Frequency of chord symbols on log-log scale.

frequently found in natural language and music [14, 15].

In order to reduce the absolute number of chord classes, several preprocessing rules were established. Chord symbols on top of a pedal, e.g. a suspended tone in the Cello, were disregarded because their harmonic function is more ambiguous. As a result, the number of chord categories was drastically reduced to only 800. Figure 1 represents the resulting distribution of chord ranks vs. chord frequencies after preprocessing.

## 2.2 N-gram Language Model

Our goal is the prediction of chord symbols, given some harmonic context. The simplest choice for a baseline model is to use an  $n$ -gram language model, which estimates the probability of the  $i^{\text{th}}$  word  $w_i$  based on the context of the previous  $n - 1$  words  $w_{i-(n-1)} \dots w_{i-1}$  as follows:

$$P(w_i | w_{i-(n-1)} \dots w_{i-1}) = \frac{C(w_{i-(n-1)} \dots w_i)}{C(w_{i-(n-1)} \dots w_{i-1})}, \quad (1)$$

where  $C(\cdot)$  counts the number of times the respective sequence of words occurs in the training data. In order to find the optimal  $n$ -gram length, hyperparameter tuning was performed. Values of  $n = 2, 3, \dots, 10$  were used to evaluate results by cross validation: for each iteration, the model was trained on the whole corpus except one opus, which was reserved for validation purposes. As a simple  $n$ -gram

Parameter	Values
Sequence Lengths [chords]	[10, 20, 40, 80, 160]
Amount of layers	[1, 2, 3, 4, 5]
Layer type	[LSTM, Bi-directional LSTM]
Amount of neurons	[8, 16, 32, 64, 128, 256, 512]
Dropout strength	[0, .1, .2, .3, .4, .5]
L2-regularization	[0, .001, .005, .01, .05, .1, .5]

Table 2. Model parameters explored

Layer	Description
LSTM	256 neurons, return sequences = True, L2 = 0
Dropout	Strength = 0.3
LSTM	64 neurons, return sequences = False, L2 = 0
Dropout	Strength = 0.3
Dense	821 neurons, activation = sigmoid, L2 = 0

Table 3. Model layout

model such as (1) can not handle unseen events, we use add-one smoothing [16] by adding one prior count to all symbols and adjusting the denominator of (1) accordingly

$$P(w_i | w_{i-(n-1)} \dots w_{i-1}) = \frac{C(w_{i-(n-1)} \dots w_i) + 1}{C(w_{i-(n-1)} \dots w_{i-1}) + V}, \quad (2)$$

where  $V$  is the total number of unique chords in the corpus.

## 2.3 Neural Network

As a more complex model for the prediction of chord symbols we used a Recurrent Neural Network (RNN) with Long Short-Term Memory cells (LSTM). This model was selected because this type of network has shown promise in sequence prediction tasks with long term dependencies [17] and thus seems suitable for an application to music. Moreover, it allows us to compare the more complex RNN model with the more basic  $n$ -gram model.

The design of the model architecture was based on related work [18, 19] after which modifications were tested manually by maximizing for validation accuracy on 10% of the data while training on the remaining 90%. Different configurations of sequence lengths, dropout, amount of layers, type of layers, and amount of neurons were tested. Tuning of the  $L_2$ -regularisation strength and dropout rate was then done with a nine-fold cross validation using the distinct opera as cross-validation folds. We tested parameters in the ranges shown in Table 2. Our final network architecture is shown in Table 3.

We also tested replacing the initial LSTM layer with a convolutional layer and performed a grid search over kernel size and amount of filters. While the training phase was notably faster, peak validation accuracy was a bit lower than our final LSTM architecture.

For the activation functions we used the defaults provided with the Keras library [20], which is tanh. To normalize the network output to a categorical distribution we used a

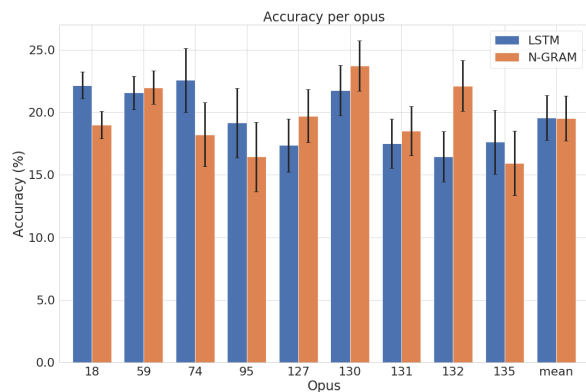


Figure 2. Comparative accuracy of a  $n$ -gram and LSTM model using nine-fold cross validation. The error bars are defined as  $1/\sqrt{n}$ ,  $n$  being the length of the opus

softmax function

$$S(x)_j = \frac{e^{x_j}}{\sum_{k=1}^K e^{x_k}}. \quad (3)$$

For training we used the Adaptive Moment Estimation (ADAM) optimizer [21], which has proven to yield robust performance based on prior work in the field of neural networks and deep learning, primarily ascribed to the adaptivity of the learning rate it employs.

### 3. RESULTS

Hyperparameter tuning for optimal  $n$ -gram length resulted in an optimal value of  $n = 2$ , which is consistent with findings in other modelling tasks in music reporting values between 2 and 4 (see e.g. [9, 16]). Thus, the simplest model actually achieved the best average accuracy score of 0.1952 (SD=0.024). While the average accuracy for  $n = 3$  and  $n = 4$  did not decrease substantially, results for larger  $n$  drastically decreased. The best recorded performance was 0.2372 for op. 130, and the lowest score of 0.1594 was achieved for op. 135. The accuracies for all opera are shown in Figure 2.

As for the LSTM model, it was observed that longer sequence lengths  $l$  in training only increased computational time at no substantial increase in accuracy, leading us to use the minimal value tested ( $l = 10$ ) for prediction. An average accuracy of 0.1958 (SD=0.026) was obtained with a maximum of 0.2257 on op. 74 and a minimum of 0.1646 on op. 132. (see Figure 2). The accuracy values for both methods and all opera are reported in Table 4. The correlation between the two model accuracies is 0.21 and thus relatively weak.

### 4. DISCUSSION

The  $n$ -gram and LSTM models have similar mean accuracies and standard deviations. The weak correlation between the  $n$ -gram and LSTM model suggests that the accuracy of the models is indicative for certain properties of

opus	LSTM	N-GRAM
18	0.2217	0.1900
59	0.2157	0.2200
74	0.2257	0.1823
95	0.1917	0.1645
127	0.1738	0.1972
130	0.2175	0.2373
131	0.1750	0.1852
132	0.1646	0.2212
135	0.1763	0.1594
mean	0.1958	0.1952

Table 4. Results

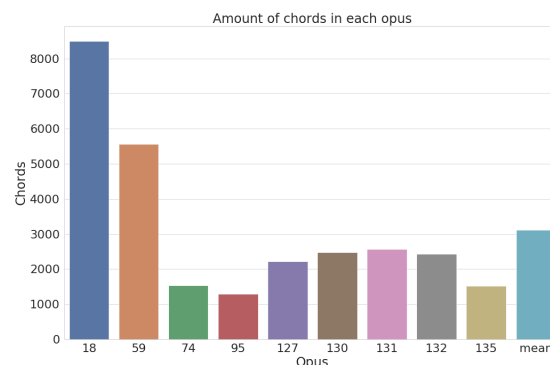


Figure 3. Amount of chords in each opus

the data. Finding out what these properties are is not only an interesting musicological research question but will also allow to improve computational models for harmony prediction in the future. Specifically, opp. 135, 95, 131, and 127 have the lowest accuracy values, which suggests that harmonic progressions within these opera are especially hard to predict.

Having a more detailed look at the performance for each opus highlights the differences (see Figure 2). For instance, for op. 95 the LSTM model demonstrates substantially better performance than the  $n$ -gram model. Op. 95 is known as one of Beethoven's most experimental works about which he stated that "this work is written for a small circle of connoisseurs and is never to be performed in public" [22]. On the other hand, in opp. 18, 74, and 135 the  $n$ -gram model outperforms the LSTM model. A better understanding of where these differences originate from is an important step and will be pursued in future research.

The best performing  $n$ -gram model was of length  $n = 2$ , which contrasts with the musicological insight that harmonic dependencies can be highly non-local. This suggests that  $n$ -gram models, which are constructed to use local context information as much as possible (even those using more advanced smoothing and backoff methods) are not able to capture long-term dependencies in harmonic progression.

LSTM models, on the other hand, are supposed to capture long-term dependencies. The fact that, overall, the LSTM model does not outperform the  $n$ -gram model on

the present data set suggests that this potential was not fully leveraged as yet. One possible reason for this might be the representation of harmonies as simple string tokens, which does not make the rich structure of the harmonic annotations in the corpus accessible to the model.

Overall, an accuracy score of 19.5% is comparably low, which is most probably due to the rich annotation format in full Roman numeral representation making the annotated Beethoven corpus a particularly challenging data set to model.

## 5. CONCLUSION

We have evaluated two of the most commonly used models for sequence prediction,  $n$ -gram models and LSTM, on a recent published data set with harmonic annotations of Beethoven string quartets (ABC). Our LSTM model and the best performing  $n$ -gram model (with  $n = 2$ ) showed comparable performance with an average accuracy of 19.5% over an alphabet of 800 harmonic symbols. The context length of  $n = 2$  suggests that neither of the models was able to pick up on non-local dependencies in harmonic progressions, which underlines the importance of incorporating structural knowledge from music theory into computational models.

As the ABC dataset is largely unexplored and is unique due to its rich annotation format, we hope that our results – especially the accuracy score of 19.5% – provide a useful baseline for other researchers in the community.

## Acknowledgments

MR would like to thank Mr Claude Latour for supporting this research.

## 6. REFERENCES

- [1] D. B. Huron, Sweet Anticipation: Music and the Psychology of Expectation. MIT press, 2006.
- [2] M. T. Pearce and G. A. Wiggins, “Auditory expectation: The information dynamics of music perception and cognition,” Topics in cognitive science, vol. 4, no. 4, pp. 625–652, 2012.
- [3] M. A. Rohrmeier and S. Koelsch, “Predictive information processing in music cognition. A critical review,” International Journal of Psychophysiology, vol. 83, no. 2, pp. 164–175, 2012.
- [4] M. Pearce and M. Rohrmeier, “Music cognition and the cognitive sciences,” Topics in cognitive science, vol. 4, no. 4, pp. 468–484, 2012.
- [5] L. B. Meyer, Emotion and Meaning in Music. University of Chicago Press, 2008.
- [6] M. M. Farbood, “A parametric, temporal model of musical tension,” Music Perception: An Interdisciplinary Journal, vol. 29, no. 4, pp. 387–428, 2012.
- [7] M. Rohrmeier and P. Rebuschat, “Implicit learning and acquisition of music,” Topics in cognitive science, vol. 4, no. 4, pp. 525–553, 2012.
- [8] J. R. Saffran, E. K. Johnson, R. N. Aslin, and E. L. Newport, “Statistical learning of tone sequences by human infants and adults,” Cognition, vol. 70, no. 1, pp. 27–52, 1999.
- [9] M. Rohrmeier and T. Graepel, “Comparing feature-based models of harmony,” in Proceedings of the 9th International Symposium on Computer Music Modelling and Retrieval. Citeseer, 2012, pp. 357–370.
- [10] F. Colombo, S. P. Muscinelli, A. Seeholzer, J. Brea, and W. Gerstner, “Algorithmic composition of melodies with deep recurrent neural networks,” arXiv preprint arXiv:1606.07251, 2016.
- [11] F. Colombo, A. Seeholzer, and W. Gerstner, “Deep artificial composer: A creative neural network model for automated melody generation,” in International Conference on Evolutionary and Biologically Inspired Music and Art. Springer, 2017, pp. 81–96.
- [12] M. Neuwirth, D. Harasim, F. C. Moss, and M. Rohrmeier, “The annotated beethoven corpus (ABC): A dataset of harmonic analyses of all beethoven string quartets,” Frontiers in Digital Humanities, vol. 5, jul 2018. [Online]. Available: <https://doi.org/10.3389/fdigh.2018.00016>
- [13] A. Graves, “Generating sequences with recurrent neural networks,” arXiv preprint arXiv:1308.0850, 2013.
- [14] S. T. Piantadosi, “Zipf’s word frequency law in natural language: a critical review and future directions,” Psychonomic bulletin & review, vol. 21, no. 5, pp. 1112–30, oct 2014.
- [15] D. H. Zanette, “Zipf’s law and the creation of musical context,” Musicae Scientiae, vol. 10, no. 1, pp. 3–18, 2006.
- [16] M. T. Pearce and G. A. Wiggins, “Improved methods for statistical modelling of monophonic music,” Journal of New Music Research, vol. 33, no. 4, pp. 367–385, 2004.
- [17] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber et al., “Gradient flow in recurrent nets: the difficulty of learning long-term dependencies,” 2001.
- [18] H. Lim, S. Rhyu, and K. Lee, “Chord generation from symbolic melody using blstm networks,” arXiv preprint arXiv:1712.01011, 2017.
- [19] S. Skuli, “How to generate music using a lstm neural network in keras,” no. Dec 7, 2017. [Online]. Available: <https://bit.ly/2IZtgm0>
- [20] F. Chollet, J. Allaire et al., “Keras,” <https://github.com/keras-team/keras>, 2019.



- [21] S. Ruder, “An overview of gradient descent optimization algorithms,” [arXiv:1609.04747](https://arxiv.org/abs/1609.04747), 2016.
- [22] B. Cooper, Beethoven. Oxford University press, 2000.

# SONIC CHARACTERISTICS OF ROBOTS IN FILMS

**Adrian B. Latupeirissa**  
Sound and Music Computing  
KTH Royal Institute of Technology  
ablat@kth.se

**Emma Frid**  
Sound and Music Computing  
KTH Royal Institute of Technology  
emmafrid@kth.se

**Roberto Bresin**  
Sound and Music Computing  
KTH Royal Institute of Technology  
roberto@kth.se

## ABSTRACT

Robots are increasingly becoming an integral part of our everyday life. Expectations on robots could be influenced by how robots are represented in science fiction films. We hypothesize that sonic interaction design for real-world robots may find inspiration from sound design of fictional robots. In this paper, we present an exploratory study focusing on sonic characteristics of robot sounds in films. We believe that findings from the current study could be of relevance for future robotic applications involving the communication of internal states through sounds, as well for sonification of expressive robot movements. Excerpts from five films were annotated and analysed using Long Time Average Spectrum (LTAS). As an overall observation, we found that robot sonic presence is highly related to the physical appearance of robots. Preliminary results show that most of the robots analysed in this study have “metallic” voice qualities, matching the material of their physical form. Characteristics of robot voices show significant differences compared to voices of human characters; fundamental frequency of robotic voices is either shifted to higher or lower values, and the voices span over a broader frequency band.

## 1. INTRODUCTION

Robots are increasingly becoming an integral part of modern society. With an increased presence of social robot interfaces comes increased demands on robots to effectively communicate with their human counterparts. The work presented in this paper is conducted within the context of the SONAO project, previously described in [1]. The SONAO project aims to improve the comprehensibility of robot non-verbal communication (NVC) through an increased clarity of robot expressive gestures and non-verbal sounds. Previous research conducted within the SONAO project has focused on developing re-targeting techniques for a NAO<sup>1</sup> robot based on findings from virtual character animation research [2] and perception of mechanical sounds inherent to expressive gestures of a NAO robot [1]. Future work in the SONAO project includes the use of

movement sonification to increase the comprehensibility of robotic gestures and emotional states. In the current study, we shift the focus from physical robots such as the NAO, to fictional robot characters in films and the sonic representations thereof. The main aim of the work is to gain insight into how Foley artists have tackled the task of designing robot sounds. We believe that findings from this exploratory work could be relevant for sound designers focusing on robotic interfaces, particularly for future implementations involving sonification of robot movements.

Previous work focusing on developing sounds for the robot NAO includes e.g. [3–5]. However, even if some previous studies have focused on sounds for communication and emotional expression in Human Robot Interaction (HRI), the sounds used in such work has often been based on simple sound synthesis methods. For example, sonification has only been used to a very limited extent in HRI (see e.g. [6, 7]). Moreover, previous work have often lacked detailed descriptions of mapping strategies or motivations of design decisions.

Our hypothesis is that sonic representations of robots in films could influence the expectations on sounds produced by real-world robots, thus affecting human robot interaction. In previous work, it has been reported that participants in sound design workshops referred to sound in movies when asked to describe sonic interaction experiences [8]. On a general note, it has been suggested that interfaces from science fiction films offer lessons to interaction designers, as science fiction interfaces reflect current interface understandings in terms of expectations from users [9], and that our concept of robots is influenced by the image of robots from science fiction [10]. In the current paper, we present an exploratory study focusing on sonic characteristics of robot sounds in films. We believe that findings from this study could be relevant for sound design in the field of HRI.

## 2. BACKGROUND

The term “*non-verbal communication*” refers to utterances that do not involve semantics in natural spoken language but may still facilitate rich communication and expression. Non-verbal communication can be organized into four categories: Gibberish Speech (GS), Non-Linguistic Utterances (NLUs), Musical Utterances (MUs) and Paralinguistic Utterances (PUs), all of which are brought together under the umbrella term Semantic-Free Utterances (SFUs) [11]. SFUs can be described as auditory communication or interaction means for machines that allow

<sup>1</sup> <https://www.softbankrobotics.com/emea/en/nao>

emotion and intend expression, composed of vocalizations and sounds without semantic content [11]. Previous research has shown that NLUs can convey affect and that people show categorical perception at a level of inferred affective meaning when listening to robot-like sounds [12]. In the current paper, we examine both non-verbal (e.g. sounds emanating from the body of the robot; robot movement sounds) and verbal (robot speech) sounds of fictional robots.

As for all products involving some kind of sound design, sounds can play a role in our aesthetic, quality, and emotional experience [13]. In [13], the authors make a distinction between sounds that are generated by operating of the product itself, and sounds that we intentionally add to a product. In the context of HRI, we need to consider both intentional sounds that are specifically designed to communicate certain emotional reactions or intentions, and consequential sounds inherent to the robot's movements. A study focusing on consequential sounds for servo motors commonly used to prototype robotic movement was presented by Moore et al. in [14]. Results suggested both anthropomorphic associations with sounds and negative impressions of motor sounds overall. In the current paper, nonverbal communication and sounds used to augment particular emotions through movement can be considered intentional sounds. One of the benefits of working with fictional robots in films is that a Foley artist can design all sounds produced by a robotic character, which is usually not the case for actual mechanical robots in real life (their movements often automatically produce sounds which are not specifically designed).

In [3], a library of emotional expressions consisting of gestures and sounds was presented. However, the authors did neither describe the sound design in detail, nor the mapping strategies used. In [4], different sounds defined to express robot emotions were evaluated using recognition ratios. Sounds ranged from gibberish speech, alienated human voices, "bleeps" to animal sounds. In [5], authors introduced BEST (Bremen Emotional Sound Toolkit)<sup>2</sup>, a validated set of 408 short (700ms to 16s) electronic sound emblems, created to augment the nonverbal capabilities of the NAO robot.

Up to this point, relatively little work has focused on how Foley artists design robot sounds. In [15], authors discuss the use of non-verbal sounds for communication of affect in interaction with robots, mentioning the sound designer Ben Burtt, who produced the sounds for the R2-D2 robot in *Star Wars* and *Wall-E*, as a source of inspiration. The story of how Burtt struggled for months before finding a R2D2 sound with credibility and character is described in detail in [16]. Burtt started experimenting with various synthesizers (Moog and ARP) to produce electronic beeps and tonalities. However, these sounds lacked emotional meaning, and Burtt therefore started blending the electronic sounds with mechanically generated sounds ("emotional" acoustic noises such as e.g. whistling sounds and expressive squeaks produced by bits of metal touch-

ing dry ice). Finally, he produced baby babble using his own voice and intercut it with electronic tones. The final version of R2-D2 involved a method in which Burtt played the synthesizer simultaneously as he recorded his voice, which in turn triggered electronic sounds and simultaneously shaped envelopes and pitches.

In [10], seven different musical sounds, five of which expressed intention and two that expressed emotion, were designed for the robot Silbot. In order to identify sound design considerations, sounds of the robots R2D2 and *Wall-E* were initially analysed. A total of 175 sound samples from *Star Wars* and 100 sounds from *Wall-E* were categorised into two different groups: intention sounds (conveying meaning/emphasizing a situation) versus emotional sounds (expressing feelings). Authors found that intonation, pitch and timbre were dominant musical parameters to express intention and emotion.

### 3. METHOD

This study aims to analyse robot sounds in films, thereby creating a basis of knowledge for future studies in the SONAO project. As mentioned above, our hypothesis is that sonic portrayal of robots in films could have an influence on expectations on sounds produced by robots. Specifically, we are looking into robot's sonic presence (i.e. sounds that signify the presence of a robot in a scene), auditory expression (i.e. sounds that signify the display of emotion), and spectral characteristics of robot speech. Results will inform the design of future sonic representation of real-world robots in the SONAO project.

#### 3.1 Film Selection

Five films were selected to be analysed in the current study. Main criteria for inclusion was that there was a presence of a humanoid robot with human-like behaviour in the film. This selection was done since the focus of the SONAO project is mainly on humanoid robotic interfaces. Furthermore, the inclusion criteria was defined so as to limit the total number of investigated robotic interfaces. The defining factor of the behaviour in this context was that the robot was capable of establishing an empathetic conversation with human characters in the film. To narrow down the selection, only non-animated films involving English-speakers were considered. In addition, it was important that the robot had sufficient screen time with no noticeable background music or noise, as the robot sounds would otherwise have to be separated from other sounds using source-separation methods. With these criteria in mind, one film was selected from each decade from the 1970s to 2010s. The selected films are *The Black Hole*<sup>3</sup> (1979, produced by Walt Disney Production); *Short Circuit*<sup>4</sup> (1986, TriStar Pictures, et al.); *Bicentennial Man*<sup>5</sup> (1999, 1492 Pictures, et al.); *I, Robot*<sup>6</sup> (2004, 20th Century Fox, et al.); and *Chappie*<sup>7</sup> (2015, MRC, et al.).

<sup>3</sup> <https://www.imdb.com/title/tt0078869/>

<sup>4</sup> <https://www.imdb.com/title/tt0091949/>

<sup>5</sup> <https://www.imdb.com/title/tt0182789/>

<sup>6</sup> <https://www.imdb.com/title/tt0343818/>

<sup>7</sup> <https://www.imdb.com/title/tt1823672/>

<sup>2</sup> <http://gaips.inesc-id.pt/emote/best-bremen-emotional-sound-toolkit/>

In the current study, all films were produced in USA, and all of the robot characters were male. The issue of representation in this context does not go unnoticed. Future studies factoring differences in culture and gender will be conducted in later stages of the current project, with work including female robot characters that fit the inclusion criteria e.g. Ava in *Ex Machina*<sup>8</sup> (2014, Universal Pictures International, et al.) and L3-37 in *Solo: A Star Wars Story*<sup>9</sup> (2018, Lucasfilm, et al.).

### 3.2 Analysis

For each selected film, video excerpts displaying respective robot and a human counterpart were isolated. The excerpts are short (ranging from 1 second to 2 minutes), containing dialogue between the characters, sonic display of emotion, or movement sound effects. Between 10 to 20 video excerpts were isolated from each film in order to be used in the analysis. All video clips that are discussed in section 4 are available as supplementary material<sup>10</sup>. For the analysis of audible sonic presence and auditory expression, each video clip was annotated and analysed from spectrograms. The results were also compared to their relation to the physical appearance and action performed by the robot. Key findings from this analysis are presented in section 4.

For the purpose of speech analysis, short video excerpts of robotic speech were isolated. For comparative purposes, video excerpts with the speech of the main human character (same gender) were also isolated. A special case was present for the film *Bicentennial Man*, where the robot, Andrew Martin (played by Robin Williams), transitioned from having a fully mechanised appearance in the beginning of the film into having a human-like appearance towards the end. In this film, we also compared the speech spectra between the robot Andrew and the human Andrew. The sound files were first analysed in Praat [17] to determine the fundamental frequency of the speech using the f0 detection scripts developed by De Looze [18]<sup>11</sup>. The sound files were then analysed using the Long-term Average Spectrum (LTAS) function `iosr.dsp.ltas` from the IoSR MatLab Toolbox<sup>12</sup> in MATLAB. Highlights of the results are presented in section 4.

By comparing speech spectra between characters from the same film, we could ensure the same quality of the sonic feature (since different films most likely will have used different approaches when it comes to sound mastering). For simplicity, the current study only focused on male characters (humans and robots). To be more precise, comparisons were made between a human (typically the main character) versus a robot in the same film.

<sup>8</sup> <https://www.imdb.com/title/tt0470752/>

<sup>9</sup> <https://www.imdb.com/title/tt3778644/>

<sup>10</sup> <https://kth.box.com/v/robotmovies>

<sup>11</sup> <http://celinedelooze.com/Homepage/Resources.html>

<sup>12</sup> <https://github.com/IoSR-Surrey/MatlabToolbox/>

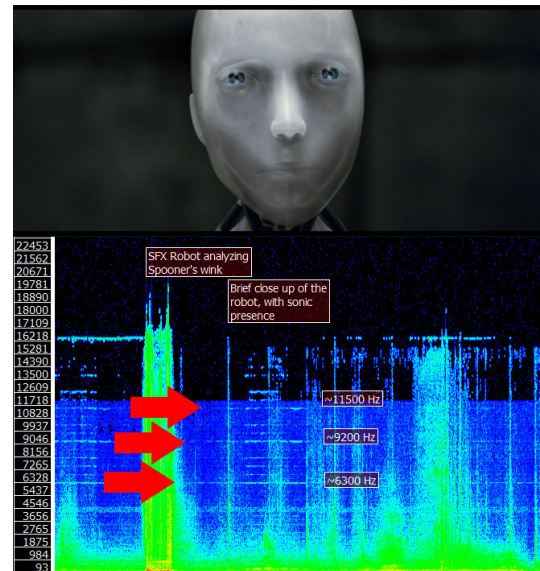


Figure 1. A close up view of the robot Sonny, accompanied by high-frequency tones.

## 4. RESULTS

An overall observation that we made after watching the films and analysing their sounds is that robot sonic presence is highly related to the physical appearance of the robot itself. Whirring sound of motors are commonly used for mechanical robots such as *Bicentennial Man*'s Andrew Martin and Chappie. These sounds are used to emphasize movements. Some of the movement sounds are also used to emphasize emotions. For example, the sound of Andrew's head movements is used to express sadness. When Andrew's head faces downwards, a mechanical sound characterized by a falling pitch is used. For Chappie, his two ears are used to emphasize his emotion; they go up or down, which is accompanied by a sound effect characterized by a rising or falling pitch.

A different approach is used for the robot Sonny in *I, Robot*. Sonny's futuristic physical appearance is much more flexible than the other robots in the current study, and this appearance is accompanied by more fluid and less mechanical sounds to emphasize his movements. Sonny's presence on the scene can be recognized by high-frequency sounds presumably emitted by his body. This is evident in the interrogation scene; as detective Spooner enters the room, the scene shows a brief close-up of Sonny's face accompanied by three simultaneous high-frequency tones centered at around 6300 Hz, 9200 Hz, and 11500 Hz respectively (see figure 1). In a later scene, where detective Spooner and Dr. Calvin enter a room to talk to Sonny, similar tones are also audible as the two human characters walk toward the robot (see figure 2). The only similar sonic presence observed for the other films in the current data set was for *Bicentennial Man*, in which the robot Andrew breaks his body after falling out of a window. This scene is accompanied by a continuous sound of broken machinery.

Analysis of the sounds of Andrew Martin as a robot ver-



Film	Character	f0-min	f0-max	Key	Range
Bicentennial Man	Andrew Martin (robot)	81	239	112	1.558
	Andrew Martin (human)	60	185	96	1.616
	Richard Martin (human)	70	293	118	2.067
Short Circuit	Number 5 (robot)	110	332	187	1.594
	Newton Crosby	125	439	199	1.808

Table 1. Highlights from speakers' register analysis from Praat: the bottom line and top line (f0-min and f0-max), Key, and Range. The f0-min, f0-max, and key are given in linear scale (Hertz), range in a logarithmic scale (octaves).

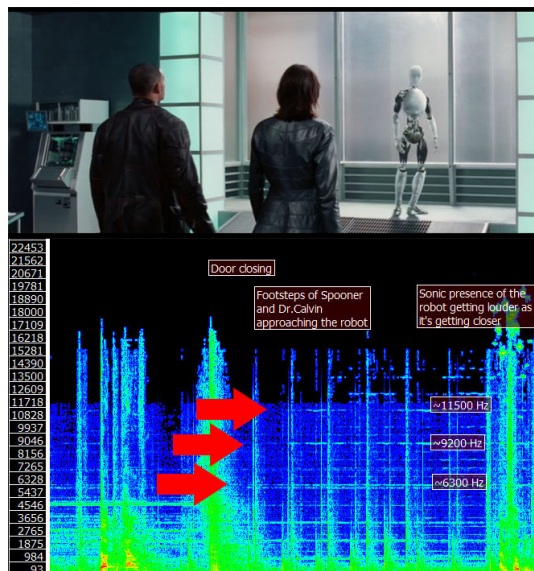


Figure 2. Similar tones are also present in other scene.

sus a human proved to be particularly interesting, based on the spectral analysis results. The human-like versus robot-like Andrew not only differed in terms of visual appearance, but also in terms of voice. Robot Andrew has a “metallic” quality in his voice, matching the material of his physical form, while the human version of Andrew retains the actor’s natural voice characteristics. Table 1 and figure 3 show significant differences between the two voices. Fundamental frequency of robot Andrew has been shifted to higher values. In addition, the robot’s voice is characterized by a broader frequency band compared to his human counterpart. In the same film, the voice of the other main human character (Richard Martin) is characterized by a narrower frequency band, compared to the robot version of Andrew (see figure 4). Similarly, in the film *Short Circuit*, the voice of the main human character (Newton Crosby) is also characterized by a narrower frequency band compared to robot Number 5 (see figure 5). The difference between *Short Circuit* and *Bicentennial Man* is that the fundamental frequency of robot character in *Short Circuit* is shifted to lower values.

## 5. DISCUSSION AND CONCLUSIONS

Of course, one may argue that other auditory features than LTAS might provide interesting information for the charac-

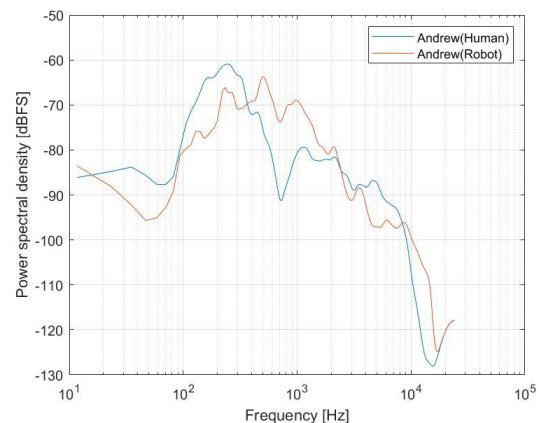


Figure 3. LTAS comparison between the two forms of Andrew Martin, human and robot.

terizations of robot sounds. Other auditory features might be of larger importance in this context, and this will be investigated in future experiments making use of voice sketching [19] for depicting robot actions and intentions. Nevertheless, we have shown that LTAS can be used for characterizing robot sounds and that robots in films are portrayed using broader frequency bands and other formants, compared to humans. Moreover, sound characteristics of the robots appear to vary both with the robot’s movements as well as its physical appearance. This information can be used in the design of future sonic renderings of robot movements and of their non-verbal sounds when interacting with humans, in combination with the manipulation of acoustical cues for rendering different emotions as shown in the research field of emotional expression in speech and music [20, 21].

For simplicity, the current study has focused only on films in which the main language was English. One may argue that sound design in HRI should be characterized by intercultural diversity, in the sense that the sounds should be interpreted similarly independently by language of origin of the listener. Still, sound design in popular films creates expectations about how a robot should sound in reality, and robots presented in films are usually associated to a specific country of origin (i.e. Japan, USA, and Germany). Therefore, the selection of films used in the present study can be considered of importance in this context. Nevertheless, as mentioned in the Method section, the number of films on which we base our analysis will be expanded in future

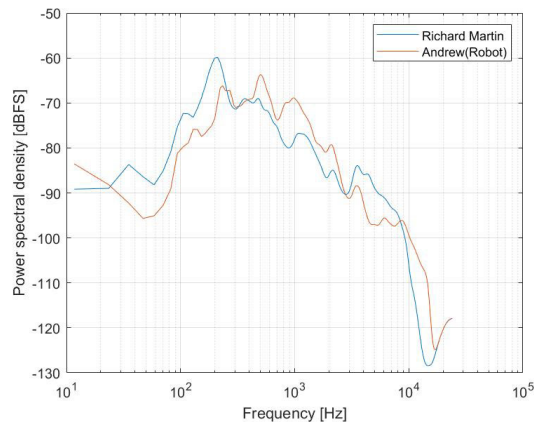


Figure 4. LTAS comparison between Richard Martin and the robot form of Andrew Martin.

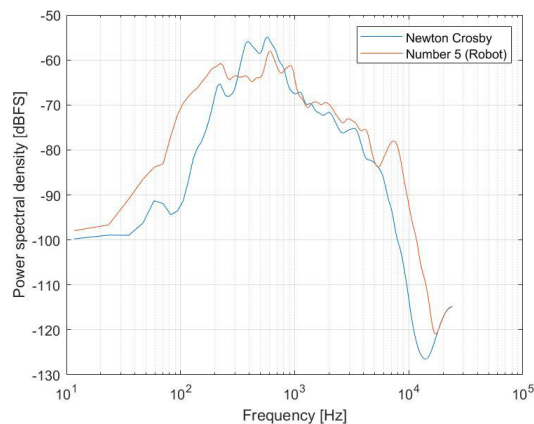


Figure 5. LTAS comparison between Newton Crosby and the robot Number 5.

work. In future data sets, female robot characters as well as robots from different countries will be represented.

### Acknowledgments

We thank the three anonymous reviewers for very helpful comments that contributed to improve the quality of our paper. This project was funded by Grant 2017-03979 from the Swedish Research Council and by NordForsks Nordic University Hub “Nordic Sound and Music Computing Network—NordicSMC”, project number 86892.

## 6. REFERENCES

- [1] E. Frid, R. Bresin, and S. Alexanderson, “Perception of Mechanical Sounds Inherent to Expressive Gestures of a NAO Robot—Implications for Movement Sonification of Humanoids,” in *Proceedings of the Sound and Music Computing Conference*, 2018, pp. 43–51.
- [2] J. B. A. Elanjimattathil Vijayan, S. Alexanderson and I. Leite, “Using Constrained Optimization for Real-Time Synchronization of Verbal and Nonverbal Robot Behavior,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, 2018, pp. 1955–1961.
- [3] J. Monceaux, J. Becker, C. Boudier, and A. Mazel, “First Steps in Emotional Expression of the Humanoid Robot NAO,” in *Proceedings of the 2009 International Conference on Multimodal Interfaces*. ACM, 2009, pp. 235–236.
- [4] M. Häring, N. Bee, and E. André, “Creation and Evaluation of Emotion Expression with Body Movement, Sound and Eye Color for Humanoid Robots,” in *Ro-Man, 2011 IEEE*. IEEE, 2011, pp. 204–209.
- [5] A. Kappas, D. Küster, P. Dente, and C. Basedow, “Simply the BEST! Creation and Validation of the Bremen Emotional Sounds Toolkit,” in *International Convention of Psychological Science*, 2015.
- [6] R. Zhang, M. Jeon, C. H. Park, and A. Howard, “Robotic Sonification for Promoting Emotional and Social Interactions of Children with ASD,” in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*. ACM, 2015, pp. 111–112.
- [7] J. Bellona, L. Bai, L. Dahl, and A. LaViers, “Empirically Informed Sound Synthesis Application for Enhancing the Perception of Expressive Robotic Movement,” in *Proceedings of the International Conference on Auditory Display*. Georgia Institute of Technology, 2017, pp. 73–80.
- [8] D. Hug, “CLTKTY? CLACK! Exploring design and interpretation of sound for interactive commodities,” Ph.D. dissertation, University of Linz, 2017.
- [9] N. Shedroff and C. Noessel, “Make it so: Learning from sci-fi interfaces,” in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, ser. AVI ’12. New York, NY, USA: ACM, 2012, pp. 7–8.
- [10] E.-S. Jee, Y.-J. Jeong, C. H. Kim, and H. Kobayashi, “Sound Design for Emotion and Intention Expression of Socially Interactive Robots,” *Intelligent Service Robotics*, vol. 3, no. 3, pp. 199–206, 2010.
- [11] S. Yilmazyildiz, R. Read, T. Belpaeme, and W. Verhelst, “Review of Semantic-Free Utterances in Social Human-Robot Interaction,” *International Journal of Human-Computer Interaction*, vol. 32, no. 1, pp. 63–85, 2016.
- [12] R. Read and T. Belpaeme, “People interpret robotic non-linguistic utterances categorically,” *International Journal of Social Robotics*, vol. 8, no. 1, pp. 31–50, 2016.
- [13] L. Langeveld, R. van Egmond, R. Jansen, and E. Ozcan, “Product sound design: Intentional and consequential sounds,” in *Advances in industrial design engineering*. InTech, 2013, p. 47.

- [14] D. Moore, H. Tennent, N. Martelaro, and W. Ju, “Making noise intentional: A study of servo sound perception,” in *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2017, pp. 12–21.
- [15] C. L. Bethel and R. R. Murphy, “Auditory and Other Non-Verbal Expressions of Affect for Robots,” in *2006 AAAI Fall Symposium Series, Aurally Informed Performance: Integrating Machine Listening and Auditory Presentation in Robotic Systems, Washington, DC, 2006*, pp. 1–5.
- [16] B. Burt, *Star Wars Galactic Phrase book & Travel Guide: Part II - Behind the Sounds*. Del Rey, 2001.
- [17] P. Boersma and D. Weenink. (2019) Praat: Doing Phonetics by Computer [Computer Program]. [Online]. Available: <http://www.praat.org/>
- [18] C. De Looze and D. Hirst, “Detecting changes in key and range for the automatic modelling and coding of intonation,” in *Speech Prosody*, 2008, pp. 135–138.
- [19] S. Delle Monache, D. Rocchesso, F. Bevilacqua, G. Lemaitre, S. Baldan, and A. Cera, “Embodied sound design,” *International Journal of Human-Computer Studies*, vol. 118, pp. 47 – 59, 2018.
- [20] P. N. Juslin and P. Laukka, “Communication of emotions in vocal expression and music performance: Different channels, same code?” *Psychological bulletin*, vol. 129, no. 5, p. 770, 2003.
- [21] R. Bresin and A. Friberg, “Emotion rendering in music: range and characteristic values of seven musical variables,” *cortex*, vol. 47, no. 9, pp. 1068–1081, 2011.

# Virtual Reality Music Intervention to Reduce Social Anxiety in Adolescents Diagnosed with Autism Spectrum Disorder

**Ali Adjorlu**

Multisensory Experience Lab  
Aalborg University Copenhagen  
adj@create.aau.dk

**Nathaly Belen Betancourt Barriga**

University of Trento  
nb.betancourtbarriga@studenti.unitn.it

**Stefania Serafin**

Multisensory Experience Lab  
Aalborg University Copenhagen  
sts@create.aau.dk

## ABSTRACT

This project investigates the potentials of Head-Mounted-Display (HMD) based Virtual Reality (VR) that incorporates musical elements as a tool to perform exposure therapy. This is designed to help adolescents diagnosed with Autism Spectrum Disorder (ASD) to deal with their social anxiety. An application was developed that combines the possibility of singing in VR while a virtual audience provides feedback. A pilot test was conducted on four adolescents diagnosed with ASD from a school for adolescents with special needs in Denmark. All four participants had shown signs of social anxiety according to their teachers. The initial results from this pilot study indicate that despite the participants' were capable of singing in front of the virtual audience without reporting a major level of social anxiety.

## 1. INTRODUCTION

Autism Spectrum Disorder (ASD) is a neurodevelopmental disorder, characterized by deficits in social communication and interaction across multiple contexts [1]. These deficits include difficulties with gestural and verbal communication, keeping eye contact and understanding facial expressions. Additionally, phobias such as social anxiety have been described to be common among individuals diagnosed with ASD since Leo Kanner first described a group of children with autism in 1943 [2,3].

Social anxiety is described as the feeling that arises in particular situations evoked by the real or imagined concern of being evaluated by others [4]. With a prevalence rate of up to 18 %, social anxiety is one of the most common psychiatric disorders while being ranked among the top 10 chronic disorders that negatively affect the general quality of life [4].

Specifically, social anxiety in children diagnosed with ASD can promote further isolation from interaction with peers, avoidance of social situations [5] and school refusal behavior [6]. These social and interpersonal problems can reduce the chances for independent adulthood. Studies show that

a vast majority of adults diagnosed with ASD are unemployed and a majority of them depend on parents or social agencies for support [7]. Furthermore, there is a correlation between anxiety, loneliness, and depression among children diagnosed with ASD [8], which has a negative effect on their general quality of life. The high prevalence of ASD (1 out of 50) further underlines the importance of developing interventions to help children and adolescents diagnosed with ASD to cope with their social anxiety [9].

Incorporating music into interventions for individuals diagnosed with ASD have illustrated benefits such as promoting engagement in social interaction and increasing self-esteem [10,11]. In the paper "Autism and Music Therapy - is change possible, and why music?" Brown argues that music is intimately interwoven to our structure as human beings and our relationship to others [12]. Even newly born infants move in synchrony with the rhythms of human voice regardless of the language [13]. Since musical elements can establish social interactions even as early as infancy, it makes sense to use these elements to children diagnosed with ASD to overcome their social anxiety.

One of the most established non-pharmaceutical methods to treat anxieties is exposure therapy [14]. By gradually exposing the patient to a stimulus that seeks to provoke anxiety (e.g., musical performance in front of an audience) without the presence of the feared outcome (e.g., being negatively judged by the audience) has proven to help individuals to overcome their phobias.

Virtual Reality (VR) can be used to create a sense of presence in a virtual environment by replacing the real-world sensory information with digitally created audio and visuals. This enables the possibility to develop controllable simulations in which therapeutic interventions such as exposure therapy can take place. Additionally, VR-based exposure therapy does not include some of the practical and logistic issues associated with real life in vivo exposure therapy. As an example, feared stimuli such as performing in front of a live audience might not be easy to access in real life while being difficult to manipulate and control (e.g., managing the audience to avoid an adverse judgment of the user).

One of the first known published studies on VR exposure therapy investigated the effectiveness of virtual environments to help individuals with agoraphobia: fear of crowded

Copyright: © 2019 Ali Adjorlu et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



places (see [15] for a review of the topic). Since then, a variety of studies have examined the effectiveness of virtual environments to conduct exposure therapy on individuals diagnosed with phobias such as acrophobia [16], zoophobia [17], and social anxiety [18]. However, there is a shortage of studies investigating the effectiveness of VR exposure therapy to help children and adolescents diagnosed with ASD to deal with their social anxiety [19,20].

Young individuals diagnosed with ASD have displayed a positive attitude towards head-mounted display (HMD) based virtual reality experiences, showing a high-level of enjoyment and immersion [21,22]. Furthermore, there have been studies illustrating the potentials of VR to teach a variety of everyday living skills to children and adolescents diagnosed with ASD [21] such as money skills [23] and social skills [24].

This study will investigate the potentials of combining the advantages of HMD-based VR exposure therapy with music and its ability to promote social interaction and connections.

## 2. METHODS

The VR intervention was designed and developed by the authors and evaluated at STUEN, a school for adolescents with mental disabilities in the Rødovre municipality.

### 2.1 The VR intervention

Traditional exposure therapy involves exposing the client to specific situations that provoke anxiety without the presence of the feared outcome. In this study, a VR application was developed to expose its users to a context within which she or he will have to sing a song in front of a virtual audience. The idea of singing in front of an audience was supported by the teachers at STUEN who informed us that some of their students with ASD had shown signs of social anxiety when they had to present a school topic in front of their peers or participate in music sessions. The application was developed using Autodesk Maya and Unity 3D, designed to run on the Oculus Rift HMD. A 3D virtual concert hall was designed containing a stage with a microphone (see Figure 1). Additionally, the virtual concert hall contained chairs for the virtual audience, a gate from which the audience would enter the hall as well as a screen (see Figure 2) on which the lyrics of the song to be performed would appear. In order to increase tension, a smaller screen was placed on the stage which was used to visualize the count down from 3 to 0 before the user had to start singing in front of the virtual audience (see Figure 1). The user was placed in front of the microphone on the stage facing the empty chairs, in order to build up the tension while providing time to familiarize herself with the virtual environment. A non-diegetic voice will then announce in Danish: "Hello. You are in the concert hall. Are you ready to make some noise? If you are ready, say come in out loud. Once you say come in, the audience will enter the hall.". The word noise is directly translated from the Danish word "larm" which means noise and is a commonly used word in Danish. Windows keyword recognizer is used

to detect the key phrase "come in" via the Oculus Rift microphone. If the user does not say "come in" after 20 seconds, the announcer will say out loud "please say come in once you are ready so we can start." Once the user has said "come in", the virtual audience will walk in through the gate of the concert hall and find their seats. Footstep sounds will further emphasize the presence of the virtual audience in the scene, having the purpose to induce the user with a bit of thrill and anxiety required in an exposure therapy intervention. A non-diegetic voice will once again announce in Danish "The audience is now seated and ready to start. Once you are ready, say start so we can begin the show". Once again, if the user does not say the keyword "start" after 20 seconds, the announcer will ask him to say "start" once more. Saying "start" will initiate a countdown from 3 to 0 which is visually presented to the user via the screen seen in figure 1 accompanied with an earcon signal for each step in the countdown. Once the countdown is over, the song will start playing, and the lyrics will be visible on the screen as seen in figure 2. The music and lyrics to be included in the VR music intervention can be changed before each session by the authors. This is done so that each user can perform their favorite song instead of a song that is chosen for them. A set of eight different virtual audience animations was developed including a variety of facial expressions and body movements (see Figure 3). These animations are triggered according to the users' singing amplitude. If the user is not singing along, the audience will sit still and look bored. If the user is singing, the virtual audience will clap their hands and look happy. If they start singing louder, the virtual audience will stand up and clap their hands. Once the song is over, the non-diegetic announcer will say in Danish "Thank you, that was awesome."



Figure 1. Screenshot from the virtual simulated room without any virtual audience. The user is placed in front of the 3D microphone. The countdown screen can be seen behind the chairs.

### 2.2 Evaluation

Four students from STUEN Rødovre diagnosed with ASD participated in this study. All participants have shown some characteristics of social anxiety. The Four participants age ranged from 18 to 20 years old and were all male. Each participant was asked if they would like to sing in front of a virtual audience in an HMD based VR application and all four accepted to participate in the study, each signing



Figure 2. The lyrics appeared on a screen behind the virtual audience.



Figure 3. The virtual audience reacting to the performance. Each member of the audience switched between eight different animations depending on the users' singing amplitude

a consent form describing the experiment. They were told that they could stop the VR session at any time. Additionally, each participant was asked which song they would like to sing, which was then added to the VR intervention by the authors before each session.

A simplified version of the Liebowitz Social Anxiety Scale (LSA) [25] was used in order to measure the extent of social anxiety in each of the participants. The LSA was simplified due to the communication difficulties observed in individuals diagnosed with ASD. The simplified version of the LSA contained only 5 items compared to the 24 items in the full version. The 5 items were chosen based on guidelines from the teachers who believed that the students would be able to understand and relate to the chosen questions. Each item in the LSA is designed to assess social anxiety in a variety of situations by asking two questions. The first question asks how anxious the participant usually is in a described situation from a scale ranging from none to mild, moderate and severe. The second question of the item asks how often the participant tries to avoid the described situation on a scale ranging from never to occasionally, often and usually. The five items included in this study are:

1. Acting, performing or giving a talk in front of an audience
2. Meeting strangers
3. Entering a room when others are already seated
4. Taking a exam

5. Looking at people you don't know very well in the eyes

All of the above situations were translated into Danish with simple and descriptive words to help the participant better understand the context. Additionally, each question was accompanied by an image to help the participant interpret the sentences. Furthermore, smiley face Likert scales (Smileyometer) were used when asking how anxious the participant usually is in the described situations as seen in figure 4. Smiley Likert scales have been reported to be effective with children. [26].



Figure 4. Smiley Likert scale for the question on how anxious the participant usually is in a described situation from a scale ranging from none to mild, moderate and severe

Finally, a timeline was designed to illustrate the answers never, occasionally, often and usually as an attempt to help the participant understand the question.

Following the simplified version of the LSA, the students were introduced to the VR application and started their task of performing a song in front of an interactive virtual audience. Data were gathered during each VR session via screen recordings of the participants' performance in the virtual environment. One author remained in the room observing the participants behavior during each session. After the VR session, the participants were asked four questions from the Witmer Singer presence questionnaire (PQ) and four questions from their Immersive Tendency Questionnaire (ITQ) [27]. This was done in order to explore whether the participants were sufficiently immersed in the virtual environment for it to be effective as an exposure therapy tool. Once again the questions were chosen based on the guidelines from the teachers and translated to Danish using simple descriptive words. The four chosen questions from the ITQ questionnaire were:

1. Do you easily become deeply involved in movies or TV dramas?
2. Do you ever become so involved in a movie that you are not aware of things happening around you?
3. How often do you play arcade or video games?
4. Do you ever become so involved in doing something that you lose all track of time?

The four questions chosen from the PQ questionnaire were:

1. How responsive was the environment to the actions that you initiated?

2. How natural did your interactions with the environment seem?
3. How involved were you in the virtual environment experience?
4. How much did the auditory aspects of the environment involve you?

The above questions on the simplified versions of the PQ and ITQ are to be answered via a 7-point Likert scale, once again communicated to the user via a smiley face Likert scale as seen in figure 5. Exposure therapy interventions

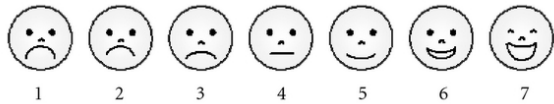


Figure 5. 7-point Smiley Likert scale for the simplified versions of the PQ and ITQ questions

expose their users gradually to a stimulus that seek to provoke the targeted anxiety without the presence of the feared outcome. Therefore, the participants were asked to rate how scary it was to sing in front of the virtual audience on a smiley Likert scale ranging from 0 to 4. In addition to the four participants, one of the teachers participated in a short unstructured interview, giving his input on the VR application as a tool to perform exposure therapy on his students. We also discussed the findings of our survey with the teacher to confirm the validity of the participants' answers.

### 3. RESULTS

All four participants tried out the VR intervention. However, P4 was not capable of understanding the questions on any of the surveys, despite them being simplified by the authors and the teacher. Therefore, only the observation data from him is presented in this paper.

#### 3.1 Liebowitz Social Anxiety Scale

The results of the simplified version of the Liebowitz Social Anxiety scale can be seen in table 1 and 2. P1 reported no fears at all from all five scenarios described to him. Furthermore, he stated that he never tried to avoid these situations. According to the teacher, this participant tries to avoid having to present in front of his classmates. P2 reported moderate fear from having to perform in front of others while he reported that he occasionally tries to avoid this kind of situations. He also stated a moderate fear from having eye contact with strangers, but he never tries to avoid it. P2 answered 'none' to the question of how much he fears meeting strangers, however, he chose occasionally when asked how often he tries to avoid situations where he has to meet new people. Entering crowded rooms is reported to be mildly feared by P2, but he indicated that he never tries to avoid having to enter them. P3 answered 'none' to how much he feared all five situations.

He also reported 'never' on how often he tries to avoid the situations in the simplified version of the LSA, except for exams which he avoids occasionally.

Participant	P1	P2	P3
Performing	None	Moderate	None
Meeting Strangers	None	None	None
Crowded rooms	None	Mild	None
Taking a exam	None	Severe	None
Eye contact	None	Moderate	None

Table 1. Responses from the simplified version of the Immersion Tendency Questionnaire asking how anxious or fearful the participant feels in different situations ranging from none to mild, moderate and severe.

Participant	P1	P2	P3
Performing	Never	Occasionally	Never
Meeting Strangers	Never	Occasionally	Never
Crowded rooms	Never	Never	Never
Exams	Never	Usually	Occasionally
Eye contact	Never	Never	Never

Table 2. Responses from the simplified version of the Immersion Tendency Questionnaire asking how often the participant avoids the situation ranging from never to occasionally, often and usually

#### 3.2 Observations during VR sessions

Each VR session lasted approximately ten minutes. P1, P3 and P4 chose to sing a Danish song called "Papirsklip" by a Danish artist called Kim Larsen. P2 chose the song "Born to be yours" by Imagine Dragons.

As previously mentioned, there were two vocal commands: "come in" which activated the animation in which the virtual audience would enter the concert hall and "start" which would initiate a countdown followed by the music starting. The users were asked to pronounce the voice commands by an announcer in the VR music intervention. P1, P2 and P3 had no problems pronouncing the commands. P4 struggled to understand the announcer and remained quiet after being asked to say "come in". The VR application was programmed to ask the user to say come in once again if the correct voice command was not detected. However, P4 remained quiet and looked shy while wearing the HMD. This resulted in the observer asking the user to say come in which was followed by P4 saying you may enter instead. Therefore, the observer had to say the voice command himself which activated the virtual audience entering the concert hall animation. P4 also struggled with the second voice command resulting in the observer once again having to activate it by saying out loud start. Once the Kim Larsen song started playing, P4 started singing along, struggling from time to time to pronounce the lyrics correctly, humming most of the song instead.

P1, P2 and P3 showed confidence when singing, looking to enjoy the experience while having no problems saying the voice commands. P2 hummed most of the song.

### 3.3 simplified version of the Immersive Tendency Questionnaire (ITQ)

The results from the simplified version of the ITQ can be seen in table 3. P1 and P2 reported a high level of immersion tendency. P3 reported that he is never so immersed in a movie that results in him not being aware of his surroundings. Furthermore, he declared that he never loses track of time when performing any activity. Questions regarding how often he plays video games and how easily he becomes deeply involved in a movie was answered with a 4 which is the midpoint on the Likert scale from 0 (never) to 7 (often).

Participant	P1	P2	P3
Easily Involved?	7	7	4
Aware of surrounding?	5	7	1
Video games	6	5	4
Loose track of time?	6	7	1

Table 3. Responses from the Immersive Tendency Questionnaire. Answer options ranged from 0 to 7.

### 3.4 Presence Questionnaire

The results from the simplified version of the PQ can be seen in table 4. All three participants reported that the VR environment was completely responsive to their actions. P1 and P3 both stated that the interaction with the VR environment seemed completely natural by choosing 7 on the smiley face Likert scale. P2 picked 6 which is just below the highest possible choice. P1 also chose the lowest

Participant	P1	P2	P3
Responsive environment?	7	7	7
Natural interaction?	7	6	7
Involvement?	6	4	7
Audio?	5	5	7

Table 4. Responses from the presence questionnaire. Answer options ranged from 0 to 7.

smiley face Likert scale out of the three participants when asked about how involved he was with the virtual environment. P1 chose 6 while P3 once again chose 7. P3 also chose 7 on whether the audio in the VR environment increased his involvement with the application. P1 and P2 both chose 5.

### 3.5 Level of anxiety

The results of the participants' answers to whether it was scary to sing in front of a virtual audience can be seen in table 5. Only P2 reported some level of anxiety. He was also the only participant to reported fearing situations where he has to perform in front of other people.

Participant	P1	P2	P3
How scary was the experience?	0	1	0

Table 5. Responses from the anxiety questionnaire. Answer options ranged from 0 to 4.

### 3.6 Teacher's comments

At the end, one of the teachers working with the participants on a daily basis tried out the VR intervention followed by a short interview with the authors. The teacher mentioned that he believed that singing and music increases what he called the fun factor of exposure therapy. However, he said that the participants he provided us with would never sing in front of other people, referring to music sessions they have had at the school. During those sessions, participants in this study often stayed away. He believed that VR might have given the illusion of being alone to his students, resulting in them singing fearlessly. When presented with the results from the Liebowitz Social Anxiety Scale, the teacher stated that his students do not want to seem scared, resulting in them answering that they do not fear the described situations. He added that P2 is probably the one with the highest level of social anxiety together with P4.

## 4. DISCUSSION AND CONCLUSION

This explorative study aimed to investigate the potentials of singing in VR as a tool to help children diagnosed with ASD to cope with their social anxieties. More specifically, the project investigates whether singing in front of a reactive virtual audience is a sufficiently immersive experience, which is one of the main requirements for exposure therapeutic interventions.

A cartoonish virtual concert hall was designed to be appealing to the user while avoiding the uncanny valley. The uncanny valley is a concept that describes the observers' revulsion towards humanoid objects or 3D models that appear nearly human [28]. Additionally, the virtual audience was programmed to react positively to the users' singing via a variety of animations such as clapping their hands and standing up while looking happy to motivate the users to sing. In contrast, if the user did not sing along, the virtual audience would remind seated, looking sad and bored. Instead of pressing a button on the VR controller to start the experience, the authors implemented audio commands to further expose the user to a situation where they had to speak in front of an audience. Finally, the participants got the option to choose the song they wanted to sing to increase their motivation.

Level of social anxiety in the participants was evaluated before each VR session using the simplified version of the LSA questionnaire, while their level of presence in the VR environment was measured using the simplified version of the ITQ and the PQ questionnaire, both developed by Witmer & Singer [27]. These questionnaires were translated to Danish together with the teachers and simplified using easy to understand words as well as illustrations visual-



ising each situation. However, P4 still had problems understanding the questions and was not capable of providing any answers. One of the primary deficits in individuals diagnosed with ASD is reduced communication capability. According to the teachers, P4 is on the low functioning side of the autism spectrum which results in him having a hard time communicating. Despite these communicative deficits, he still sang and hummed during the VR session, activating positive feedback from the virtual audience. This behavior was also observed in P1, P2 and P3 who sang or hummed during the VR session, activating the virtual audience's positive feedback. The singing behavior of the participants was surprising to the teacher, who did not expect them to behave the way they did. However, according to the results from the simplified version of the LSA questionnaire, P1 and P3 reported having no stage fear, making their performance on the virtual stage less surprising. The teacher's comment on this result is that they did not want to be perceived negatively by the authors by stating that they were scared of certain situations. In general, the results from the simplified version of the LSA does not correlate with the comments from the teacher who stated that his students suffer from social anxiety. Therefore, the validity of the simplified version of the LSA as well as the PQ, and ITS questionnaires is debatable. In future iterations, a qualified expert clinician or behavioral psychologists should be involved in the design of the experiment, helping to gather more valid data from this target group. Additionally, to gain further information on the social characteristics of the participants, methods such as the Social Responsiveness Scale could be used [29].

In the post VR session questionnaires, P2 rated 4 on his self reported involvement in the VR experience, which was the lowest of all three participants. This low level of involvement with the VR music intervention can explain why he was singing along despite his moderate fear of performing in front of an audience (according to his teacher). He also rated 6 on whether the interaction in the VR environment was natural, which is also the lowest rating from the three participants. He was the one who hummed to most of the song compare to the other participants who sang the words which can be explained by the fact that he was the only one who chose an English song to sing. P2 was not asked any question on whether he was able to speak English. In future iterations, only danish songs should be included in the VR application. The teacher suggested the songs included in this version of the application after he had asked the participants which songs they would like to sing in VR.

P3 choose the smiley face Likert item number 7, the highest possible on all four questions on the simplified version of the PQ survey. This is despite him reporting the lowest immersive tendency out of the three participants who answer the surveys. During the singing session, he was the one who seemed to know most of the lyrics of the Kim Larsen song, and also seemed to enjoy himself. Therefore, the music could explain his high rating in the simplified version of the PQ survey.

P1 reported being easily involved in movies and loses

tracks of time when performing certain activities. He also reported a high level of presence in the VR music intervention, which was also observed during his singing session. He sang the lyrics, putting effort into trying to read the lyrics and pronouncing them correctly.

P4 was not capable of answering neither the simplified versions of the LSA, the ITQ or the PQ surveys, even though they were simplified and had images explaining their content. The teacher explained that P4 has the lowest communication skills of all the participants. He also had problems understanding the announcer in the VR experiment which asked him to repeat the voice commands "come in" and "start.". However, when the music started playing, he did not hold back and started to sing along, correctly pronouncing the lyrics of the Danish song he chose. Music might have reduced the social barriers he experienced since he looked relaxed during the VR singing session. Future iteration of this study could involve the teacher gathering data from his students about their experience with the VR music intervention. The students feel more comfortable with their teachers. This might result in them providing more relevant and valuable insight about their experience with the VR intervention. Using the simplified version of the LSA, PQ, ITQ was an attempt to not overwhelmed the users with a lot of questions. However, doing this we have reduced the validity of these methods.

Despite of this, we believe that the explorative study showed that a VR music intervention could be an immersive experience for adolescents diagnosed with ASD and social anxiety. As Brown argued, music seemed to remove barriers for P4 who started singing along the song moments after he was timid and unable to communicate with the observer [12]. In future iterations of the study, information on the participants' reaction to the feedback provided by the virtual audience must be collected to shed some light on its effectiveness to increase the users' immersion.

Exposure therapy is defined by gradually exposing the user to a stimulus that provokes anxiety without the presence of the feared outcome. Out of the three participants who answered questions after the study, only P2 reported experiencing any form of anxiety. Future iterations of the application should focus on creating a more tense experience to provoke more social anxiety in its users by adding new levels in the VR application. The first level will keep the cartoonish background and 3D virtual audience. The next level will replace the environment with a real classroom or concert hall recorded via a 360 camera while keeping the virtual audience. The third level will only consist of 360 video footage of real audience and environment recorded with 360 camera, making it as realistic as possible.

## Acknowledgments

Special thanks to Sune Buch-Sloth from Rødovre Municipality who put us in contact with STUEN Rødovre. We would also like to thank Thor Jønsson, the teacher at the STUEN Rødovre who helped us find relevant participants for the study and used his time to provide us with feedback.

## 5. REFERENCES

- [1] A. P. Association *et al.*, *Diagnostic and statistical manual of mental disorders (DSM-5®)*. American Psychiatric Pub, 2013.
- [2] A. Gillott, F. Furniss, and A. Walter, "Anxiety in high-functioning children with autism," *Autism*, vol. 5, no. 3, pp. 277–286, 2001.
- [3] D. Spain, J. Sin, L. Harwood, M. A. Mendez, and F. Happé, "Cognitive behaviour therapy for social anxiety in autism spectrum disorder: a systematic review," *Advances in Autism*, vol. 3, no. 1, pp. 34–46, 2017.
- [4] M. B. Stein and D. J. Stein, "Social anxiety disorder," *The lancet*, vol. 371, no. 9618, pp. 1115–1125, 2008.
- [5] B. S. Myles, G. P. Barnhill, T. Hagiwara, D. E. Griswold, and R. L. Simpson, "A synthesis of studies on the intellectual, academic, social/emotional and sensory characteristics of children and youth with asperger syndrome," *Education and Training in Mental Retardation and Developmental Disabilities*, pp. 304–311, 2001.
- [6] E. K. Munkhaugen, E. Gjevik, A. H. Pripp, E. Sponheim, and T. H. Diseth, "School refusal behaviour: Are children and adolescents with autism spectrum disorder at a higher risk?" *Research in Autism Spectrum Disorders*, vol. 41, pp. 31–38, 2017.
- [7] D. Hendricks, "Employment and adults with autism spectrum disorders: Challenges and strategies for success," *Journal of Vocational Rehabilitation*, vol. 32, no. 2, pp. 125–134, 2010.
- [8] M. O. Mazurek, "Loneliness, friendship, and well-being in adults with autism spectrum disorders," *Autism*, vol. 18, no. 3, pp. 223–232, 2014.
- [9] G. Xu, L. Strathearn, B. Liu, and W. Bao, "Prevalence of autism spectrum disorder among us children and adolescents, 2014-2016," *Jama*, vol. 319, no. 1, pp. 81–82, 2018.
- [10] S. M. Shore, "The language of music: Working with children on the autism spectrum," *Journal of Education*, vol. 183, no. 2, pp. 113–124, 2002.
- [11] D. Wimpory, P. Chadwick, and S. Nash, "Brief report: Musical interaction therapy for children with autism: An evaluative case study with two-year follow-up," *Journal of autism and developmental disorders*, vol. 25, no. 5, pp. 541–552, 1995.
- [12] S. M. Brown, "Autism and music therapy is change possible, and why music?" *Journal of British music therapy*, vol. 8, no. 1, pp. 15–25, 1994.
- [13] W. S. Condon and L. W. Sander, "Neonate movement is synchronized with adult speech: Interactional participation and language acquisition," *Science*, vol. 183, no. 4120, pp. 99–101, 1974.
- [14] A. M. Reid, M. I. Bolshakova, A. G. Guzik, A. G. Fernandez, C. W. Striley, G. R. Geffken, and J. P. McNamara, "Common barriers to the dissemination of exposure therapy for youth with anxiety disorders," *Community mental health journal*, vol. 53, no. 4, pp. 432–437, 2017.
- [15] A. Adjorlu, "Virtual reality therapy," in *Encyclopedia of Computer Graphics and Games (ecgg)*. Springer, 2018.
- [16] P. M. Emmelkamp, M. Krijn, A. Hulsbosch, S. De Vries, M. J. Schuemie, and C. A. van der Mast, "Virtual reality treatment versus exposure in vivo: a comparative evaluation in acrophobia," *Behaviour research and therapy*, vol. 40, no. 5, pp. 509–516, 2002.
- [17] C. Botella, M. Á. Pérez-Ara, J. Bretón-López, S. Quero, A. García-Palacios, and R. M. Baños, "In vivo versus augmented reality exposure in the treatment of small animal phobia: a randomized controlled trial," *PloS one*, vol. 11, no. 2, p. e0148237, 2016.
- [18] E. Carl, A. T. Stein, A. Levihn-Coon, J. R. Pogue, B. Rothbaum, P. Emmelkamp, G. J. Asmundson, P. Carlbring, and M. B. Powers, "Virtual reality exposure therapy for anxiety and related disorders: A meta-analysis of randomized controlled trials," *Journal of anxiety disorders*, vol. 61, pp. 27–36, 2019.
- [19] D. E. Parrish, H. K. Oxhandler, J. F. Duron, P. Swank, and P. Bordnick, "Feasibility of virtual reality environments for adolescent social anxiety disorder," *Research on Social Work Practice*, vol. 26, no. 7, pp. 825–835, 2016.
- [20] R. Bradley and N. Newbutt, "Autism and virtual reality head-mounted displays: a state of the art systematic review," *Journal of Enabling Technologies*, vol. 12, no. 3, pp. 101–113, 2018.
- [21] N. Newbutt, C. Sung, H.-J. Kuo, M. J. Leahy, C.-C. Lin, and B. Tong, "Brief report: A pilot study of the use of a virtual reality headset in autism populations," *Journal of autism and developmental disorders*, vol. 46, no. 9, pp. 3166–3176, 2016.
- [22] D. Strickland, L. M. Marcus, G. B. Mesibov, and K. Hogan, "Brief report: Two case studies using virtual reality as a learning tool for autistic children," *Journal of Autism and Developmental Disorders*, vol. 26, no. 6, pp. 651–659, 1996.
- [23] A. Adjorlu and S. Serafin, "Head-mounted display-based virtual reality as a tool to teach money skills to adolescents diagnosed with autism spectrum disorder," in *Interactivity, Game Creation, Design, Learning, and Innovation*. Springer, 2018, pp. 450–461.
- [24] A. Adjorlu, A. Hussain, C. Mødekjær, and N. Warming Austad, "Head-mounted display-based virtual reality social story as a tool to teach social skills to children diagnosed with autism spectrum disorder," in

*2017 IEEE Virtual Reality Workshop on K-12 Embodied Learning through Virtual & Augmented Reality (KELVAR)*. IEEE, 2018.

- [25] R. G. Heimberg, K. Horner, H. Juster, S. Safren, E. Brown, F. Schneier, and M. Liebowitz, "Psychometric properties of the liebowitz social anxiety scale," *Psychological medicine*, vol. 29, no. 1, pp. 199–212, 1999.
- [26] J. C. Read, S. MacFarlane, and C. Casey, "Endurability, engagement and expectations: Measuring childrens fun," in *Interaction design and children*, vol. 2. Shaker Publishing Eindhoven, 2002, pp. 1–23.
- [27] B. G. Witmer and M. J. Singer, "Measuring presence in virtual environments: A presence questionnaire," *Presence*, vol. 7, no. 3, pp. 225–240, 1998.
- [28] M. Mori, K. F. MacDorman, and N. Kageki, "The uncanny valley [from the field]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 98–100, 2012.
- [29] J. N. Constantino, S. A. Davis, R. D. Todd, M. K. Schindler, M. M. Gross, S. L. Brophy, L. M. Metzger, C. S. Shoushtari, R. Splinter, and W. Reich, "Validation of a brief quantitative measure of autistic traits: comparison of the social responsiveness scale with the autism diagnostic interview-revised," *Journal of autism and developmental disorders*, vol. 33, no. 4, pp. 427–433, 2003.

# Teach Me Drums: Learning Rhythms through the Embodiment of a Drumming Teacher in Virtual Reality

**Mie Moth-Poulsen**  
Aalborg University Copenhagen  
miemp123@gmail.com

**Tomasz Bednarz, Volker Kuchelmeister**  
UNSW Art and Design, Sydney  
t.bednarz, Kuchel@unsw.edu.au

**Stefania Serafin**  
Aalborg University Copenhagen  
sts@create.aau.dk

## ABSTRACT

This paper investigates how to design an embodied learning experience of a drumming teacher playing hand drums, to aid higher rhythm understanding and accuracy. By providing novices the first-person perspective of a drumming teacher while learning to play a West-African djembe drum, participants' learning was measured objectively by their ability to follow the drumming teachers rhythms.

Participants subjective learning was assessed through a self assessment questionnaire measuring aspects of flow, user-experience, oneness, and presence. Two test iterations were conducted. In both there was found no significance difference in participants' ability to follow the drumming teacher's tempo for the experimental group exposed to the first-person perspective of the teacher in a Virtual Reality (VR) drum lesson, versus the control group exposed to a 2D version of the stereoscopic drum lesson. There was found a significant difference in the experimental group's presence scores in the first test iteration, and a significant difference in experimental group's oneness scores in the second test iteration. Participants' subjective feelings indicated enjoyment and motivation to the presented learning technique in both groups.

## 1. INTRODUCTION

Several studies have shown the potential of Virtual Reality (VR) as an alternative training and learning platform for acquiring new skill sets and improving existing ones. Amongst others seen in the field of music learning, training rhythmical skills and musical expression [1], along training physical movements [2]. The multidimensional nature of VR provides a unique possibility to take the perspective of another person than one self. This quality provides a strong tool to facilitate learning and communication between individuals, which has been examined by expert-novice mentorship simulations in the art of painting [3]. One advantage of using VR for education is the ability to present abstract topics in a tangible way. For example, teaching mathematics through collaborative environments for learning geometrical concepts, against traditional pen



Figure 1. First drumming recordings.



Figure 2. Second drumming recordings.

and paper. Moreover, VR supports doing, instead of observing, as the user participates in the virtual world instead of using it, compared to other types of human-computer interfaces [4]. The focus on incorporating the body in designing movement-based interfaces has been fueled by the advances in sensor technology [5]. The affordance of VR technology leverages interactivity, not seen in medias such as video and text, based on the vast possibilities for behavioral tracking. This essentially allows users to participate in a VR instead of using it, through embodied interactions. This project seeks to investigate how to teach music more effectively, by incorporating VR to communicate somatic knowledge of a drumming teacher, providing a first-person perspective of the playing teacher.

Copyright: © 2019 Mie Moth-Poulsen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.





Figure 3. A screenshot from the experiment. Top: experimental condition: wearing the VR headset. Bottom: control condition where the visual feedback is given by the screen. In both cases auditory feedback is given by headphones, while tactile feedback is given by the real drum.

## 2. DESIGN AND IMPLEMENTATION

This section presents the design of a VR setup for teaching novices drums through an embodied first-person perspective of a drumming teacher. The underlying motivation is to aid higher rhythm understanding and accuracy through the participants embodiment of the playing teacher. The design consists of two parts. First, the development of the teaching material constituting the test stimuli of a prerecorded drum lesson, and a discussion of the suitable hand drum for the rhythm teaching of novices. Secondly, the recording setup with a stereoscopic 3D camera is discussed, for capturing a reliable first-person perspective of the drumming teacher, to be viewed through a Head Mounted Display (HMD) while playing along the teacher.

### 2.1 Design of Teaching Material

The design of the teaching material for the drumming recording was revised and discussed in two iterations, with two professional drummers respectively. A prerequisite to the drummers' qualifications was teaching experience and skills on hand drums of a West-African Djembe and a bongo drum. Hand drums were chosen based on perceived affordances and signifiers on how to be operated by the hands. Secondly, allowing immediately tactile feedback, to provide a direct sense of the drum when viewing the stereoscopic footage of the hand drum through a HMD.

Before each of the two individual drumming recordings, the drummers were explained motivation for the research

of the study. Additionally, the test objective of comparing two learning medias (VR and video), and their effectiveness on rhythm learning. The requirements for the teaching material included content directed to novices who had no to little experience with drum lessons before. This was to ensure an equal level of drumming experience, and skill level between the test participants. Furthermore, the drum recording was structured with drumming sequences that left enough time for the novice to play along. This is based on the objective measure, to test the effect of the two learning medias, which involves a comparison of the drumming rhythms produced by the mentor and the novice. The drummers' knowledge was incorporated into structuring the suitable rhythms for teaching hand drums to novices.

The first drummer had over 40 years experience of music practicing. The teachers experience included teaching, performing drumming shows with a west African drumming group, along skills in other instruments. A bongo drum (size 6 and 7 inch) was used for the first recording (see Figure 1), due to affordability. An initial instruction and trial phase was dedicated to familiarization with the bongo before the rhythm training. It was incorporated in the first part of the drum lesson, allowing the participants to get familiar with the bongo. This included, how to place the bongo between the knees, the hit method, and how to produce a pulse on the bongo. The general structure of the drumming lesson was composed of 4 sequences of rhythm patterns. The teaching material for both drumming recording followed the general structure of a trial phase and four rhythm patterns. The trial phase included how to hit the djembe. The teacher instructed how to play two different djembe tones in the trial phase, to match the skill level for a novice. The base tone (centre drum skin), and the tone (edge of drum skin). Halfway in each play along sequence, the teacher increased the tempo, to challenge the novice.

### 2.2 Recording setup

The recording was filmed with the stereoscopic 3D Lucid-Cam. For the second drumming recording, the static rig was improved from the first recording. The rig was altered to be positioned from the side, without the two fish-eye lenses capturing the extended rig arm, allowing more flexibility for adjusting to the drummers height (see Figure 2).

### 2.3 Implementation

The game engine Unity3D 2017 was used as the software to implement the 3D stereoscopic viewing for the HTC Vive HMD. The digital hands from a Leap Motion device were incorporated to support the position of novices own hands within the physical drumskin, and the matching of their own hands to the playing teachers.

## 3. FIRST ITERATION

The first test iteration included a between-group design, comparing two viewing conditions of VR and video. The test stimuli for both test groups were the same pre-recorded video of a drum lesson instructed by a drumming teacher, from the second drum recording. The test stimuli consisted

of 5 phases. An initial trial phases taught the participants how to hit the djembe drum, to get comfortable with it. Next phases four rhythm patterns were taught. In each rhythm pattern, the teacher demonstrated the rhythm three times on the djembe drum, before playing along. Each play along rhythm sequence was on average 49.5 seconds long. The total length of the test stimuli was 5 minutes and 53 seconds. The first 5 seconds of the recordings was a black screen. 11 seconds was left in the end, after the teacher finished the final and fourth rhythm. In the experimental group, participants were taught to play drums by taking the first-person perspective of the drumming teacher in VR, viewed through a HMD, projecting the stereo-scopic recording of the teacher. Participants were presented with a physical djembe drum, matched to the position where it was located in the virtual world (test stimuli). In the control group, participants viewed the same test stimuli on a 2D monitor placed in front of them. The audio of the test stimuli was recorded with the built in stereo microphones in the LucidCam, recording audio at 48Hz, uncompressed 16-bit audio, from the teachers visual perspective. The viewed test stimuli was recorded in the same room as the participants were seated, with the participants sitting on the same position as the teacher on a stationary chair.

### 3.1 Participants and Recruitment

35 participants were recruited at the University of New South Wales Art Design (Sydney). The data of five participants was not usable and discarded, producing a final sample of 30 (male=12, female=18). The groups ages ranged from 20 to 40; the majority accounted 25-34 years (50%). The majority were students (86.6%). Before conducting the test, participants were handed a consent form along with a participant information sheet for the test. Participants' data was assured confidentiality, along with their right to withdraw from the test at any time. Participants were recruited based on the criterias of being novices to drumming, not having any hearing disabilities, and fully functional limbs.

### 3.2 Setup

A djembe hand drum of height 60cm and diameter 30cm was used. Participants were seated on a stationary stool of fixed height 43.5 cm, and a diameter of 33 cm. The drumming lesson took place in a section of a closed room, partly covered by a black curtain. Each participant's sound from the drumming, was recorded with a lavalier microphone clipped to their clothes, centre at chest. On-ear headphones were used to play the sound of the viewed test-stimuli. The HTC Vive HMD was used to display the test stimuli for the experimental group. The control group sat at a distance of one meter from the 2D monitor.

### 3.3 Procedure

Participants were randomly assigned to each of the two test groups, and all naive to the purpose of the test. Participants were asked demographic questions before the drumming lesson in each of the two test groups (experimental

and control), in an online self-assessment questionnaire. Both groups were asked about their experiences with music practising. The experimental group was further asked about their previous experience with VR, by a rating scale from 1 (never tried it before) to 7 (uses it daily). Participants were recorded to compare the rhythm accuracy between the drumming teachers recording and the participant. At the beginning of the viewed test footage, an audible synchronisation clap from the test footage, was outputted through a pair of speakers, before switching the sound output to the worn headphones by the participant. This was done to generate the same synchronization point in the participants audio files to the teacher, for the rhythm comparison. After the pre-recorded drum lesson, participants completed post questions to their subjective learning experience, filled out in the online self-assessment questionnaire.

### 3.4 Rhythm accuracy

The variable of rhythm accuracy was chosen to objectively quantify the novices ability to follow the teachers djembe rhythms. Each participant produced a unique audio file, capturing their rhythmic performance during the experiment. The learning efficiency of both test groups was quantified by comparing participants individual audio files analysed in beats by minute, with the drumming teachers.

### 3.5 Self-assessment questionnaire

Participants answered a self-assessment questionnaire post to their participation in the pre-recorded drumming lesson. The questionnaire was designed to measure 4 aspects of the participants subjective experience of the given learning media, including: Flow, User-experience, Oneness, and Presence. Flow was measured using the Flow Short Scale [6].

The Inclusion of Other in the Self Scale was originally developed in [7]. The scale was used to measure the extent to which the participants felt bodily in sync with the mentors location and rhythmic movements, related to the concept of body-syntonicity. The measure of presence targeted presence seen through the Plausibility Illusion (Psi). Psi relates to the fact that the scenario presented is felt as actually happening [4].

In the experimental group, the audio recordings of five participants was discarded. Participants noticed that the audio and video played through the Unity application was out of sync. The synchronization of the audio and video played through Unity was therefore monitored during each test, by the participants wearing the HMD at their drumming position before starting and playing of the unity application. Participants soundfiles were synchronized with the teachers in post-processing. The drumming lesson consisted of four rhythm patterns, both including a normal and fast tempo. Eight soundfiles were generated for each participant, containing the normal and fast tempo of each rhythm. It was prioritized to avoid the teachers voice in the generated soundfiles, not to cause a further difference in the teachers and participants audio recordings. Eight sequence markers of label tracks were created related to

the teachers starting points. The sequence markers contained both the normal and fast tempo of the rhythm play along periods. The sequence markers ensured the generation of the eight individual wav files for each participant, to match precisely the teachers soundfiles files respectively. The MIRtoolbox 1.7 for Matlab, was used to analyse participants rhythm performances [8].

An independent t-test was performed with participants final tempo score in the two test groups. The reported results showed no significance difference for the experimental group exposed to the VR drum lesson ( $M=35.311$ ,  $SE=2.956$ ), than for the control group exposed to the 2D drum lesson ( $M= 30.821$ ,  $SE=2.1043$ ),  $t(28)= 1.236$ ,  $p= 0.226$

Though there was found no significant difference among the two test groups, the control group performed better with 4.49 BMP less in difference from the drumming teacher, than the experimental group. Inspecting the data further, the total average of the fast tempo difference scores of the two test groups, produced almost equal results. The control group produced a fast tempo difference score with a mean value of 20.29 BPM. While the experimental group produced a mean value of 20.49 BPM. The total average of the normal tempo difference scores produced 41.36 BPM in the control group, and 50.13 BMP in the experimental group, indicating the control group was 8.77 BMP closer to following the teachers tempo, than the experimental group, in these sequence parts. An independent t-test was also performed among the two groups total average of the normal tempo scores. The result also indicated no significance among the two groups ability to follow the teachers tempo in normal pace. An independent t-test was performed with participants average flow score from the FSS in the two test groups. The result of the FFS flow scores was found not significantly different for the experimental group exposed to the VR drum lesson ( $M=5.580$ ,  $SE=0.152$ ) than for the control group exposed to the 2D drum lesson ( $M= 5.1067$ ,  $SE=0.254$ ),  $t(28)= 1.601$ ,  $p= 0.121$ .

Participants' ratings in the first seven user-experience items produced a total mean value for the experimental group at 5.73, and a mean value at 5.70 for the control.

There was found no significance difference between the experimental and control group, in the reported oneness ratings, with the exact same mean of 4.73, and median of 5 in both groups. The Oneness scores indicated that the first-person perspective rendered in the HMD experienced by the experimental group did not provide a stronger sense of feeling in tune and synchronizing with the teachers movements in this experimental setup.

An independent t-test was performed with participants average presence score in the two test groups. The result of the presence scores was found significantly different for the experimental group exposed to the VR drum lesson ( $M=5.617$ ,  $SE=0.251$ ) than for the control group exposed to the 2D drum lesson ( $M= 4.7333$ ,  $SE=0.263$ ),  $t(28)= 2.4282$ ,  $p= 0.0219$ ,  $= 0.05$ . Table 1 summarizes the results.

Measurement	Mean	Std. dev.	Std. err.
Flow Exp.	5.580	0.588	0.152
Flow Con.	5.1064	0.982	0.254
User-experience Exp.	0.833		0.215
User-experience Con.	0.827		0.213
Oneness Exp.	4.733	1.792	0.463
Oneness Con.	4.733	1.624	0.419
Presence Exp.	5.617	0.972	0.251
Presence Con.	4.733	1.019	0.263

Table 1. Descriptive statistics of the Subjective Self-assessment Questionnaire for each measurement for the control (con) and experimental (exp) group for the first test iteration.

## 4. SECOND ITERATION

The experimental design followed the same as the first test iteration. However, the test's stimuli was revised with a third drumming recording, to produce a final pre-recorded drum lesson. The hired drumming teacher was the same used for the test stimuli in the first test iteration. The structure of the test stimuli followed the same as the first test iteration, with the same four rhythms. The total length of the recorded test stimuli was 7 minutes and 9 seconds long. The test stimuli consisted of a longer trial phase with more deliberate instructions. This was to ensure that the participants got a sense of how to hit the djembe properly before the first rhythm instruction. The four rhythm patterns taught was on average 82 seconds long. The audio recording of the mentor in the first experiment was recorded with the built-in stereo microphones in the LucidCam. In the first experiment, the participants could see what the teacher saw, but not hear a reliable version of what the teacher heard. This experiment revised the audio capture to be recorded from a binaural point-of-view with the Roland CS-10EM binaural microphones. The microphones are electret and omnidirectional, capturing a frequency range of 20 Hz to 20,000 Hz.

### 4.1 Participants and recruitment

41 participants were recruited at the University of New South Wales, Art Design (Sydney). It was ensured that none of the subjects had participated in the first test iteration. One participants data was not usable and discarded, giving a final sample of 40 (male=14, female=26). The groups ages ranged from less than 20 to 55; the majority accounted less than 20 years (35%) and 21 – 25 years (30%). The majority were students (80%). Participants were recruited by the same conditions as in the first test iteration.

### 4.2 Setup and procedure

The setup and procedure followed the general structure of the first test iteration. The sound of the participants was captured with a Zoom H4n Pro, placed in front of the drum. The changes to the procedure included the test conductor demonstrating how to hold and position the djembe cor-

Measurement	Mean	Std. dev.	Std. err.
Flow Exp.	5.333	1.159	0.259
Flow Con.	5.105	0.881	0.197
User-experience Exp.	5.714	1.061	0.237
User-experience Con.	5.728	0.828	0.185
Oneness Exp.	5.250	1.564	0.336
Oneness Con.	4.4	1.353	0.303
Presence Exp.	5.575	1.162	0.259
Presence Con.	4.975	1.186	0.265

Table 2. Descriptive statistics of the Subjective Self-assessment Questionnaire for each measurement for the control (con) and experimental (exp) group for the second test iteration.

rectly, with the right angle tilting the djembe from the floor away from the participant. Furthermore, the test conductor explained in the brief about the drum lesson content, that the participant would first be instructed by the teacher in the drum lesson on how to hit the djembe. Next, following a count in on four beats, to hit along the teacher. The evaluation of the second test iteration relied on the same measurements methods used in the first test iteration.

### 4.3 Results

Similarly to the first test iteration, there was found no significant difference between the experimental group exposed to the VR drum lesson ( $M = 24.567$ ,  $SE = 2.282$ ), and the control group exposed to the 2D drum lesson ( $M = 21.739$ ,  $SE = 1.932$ ),  $t(38) = 0.946$ ,  $p = 0.350$ .

### 4.4 Subjective learning

There was found no significance in the subjective ratings of flow, user-experience, and presence. The Mann-Whitney test was used as a significance test of the oneness ratings, due to non-parametric data. The feeling of oneness with the teacher showed significance difference between the experimental group exposed to the VR drum lesson ( $M = 5.250$ ,  $SE = 0.336$ ), than for the control group exposed to the 2D drum lesson ( $M = 4.400$ ,  $SE = 0.303$ ),  $t(38) = 1.558$ ,  $p = 0.048$ . Table 2 summarizes the results.

## 5. CONCLUSIONS

This paper investigated if learning hands drums through an embodied first-person perspective mediated in VR leads to better rhythmic understanding than learning through a 2D video. The results of the rhythm comparison in the two test iterations found no significant difference between the experimental and control group learning of rhythms, evaluated in the ability to follow the teachers tempo in BPM. The majority of the participants described their experience as enjoyable in both test iterations. Additionally, indicating motivation towards the given learning technique in both test groups. The results can situate the question whether the given musical instrument and task was a motivation, along the given teaching material. The two test conditions

were designed to detect the effect of a first-person perspective of a drumming teacher, on a novices rhythm accuracy and learning. Thus, the two conditions differed in visual display, the control group had the ability to watch their own hands playing on the physical djembe. This could produce an advantage in terms of acquiring a better sense of the edge when the hitting drum skin. However, a restrains in this scenario was the shifting of attention between the participants hands and the playing teacher viewed on the 2D monitor in front of them. In a first iteration, we tried to project the hands of the player of top of the VR experience, using the Leap Motion's tracking. However, the inconsistent tracking of the participants hands caused the attention to be directed to the quality of the 3D rendered hands more than the experience. As the viewed drum lesson was a pre-recorded video, the possibility of corrections to participants rhythm performance real-time from a teacher was not available. Participants were not given any feedback upon how well they performed the rhythm in the first and second test iteration - from an objective source. To optimize the participants learning, future studies could explore real-time feedback of participants rhythm performances.

## 6. REFERENCES

- [1] S. Serafin, A. Adjorlu, N. Nilsson, L. Thomsen, and R. Nordahl, "Considerations on the use of virtual and augmented reality technologies in music education," in *K-12 Embodied Learning through Virtual & Augmented Reality (KELVAR)*, 2017 IEEE Virtual Reality Workshop on. IEEE, 2017, pp. 1–4.
- [2] J. Bailenson, K. Patel, A. Nielsen, R. Bajscy, S.-H. Jung, and G. Kurillo, "The effect of interactivity on learning physical actions in virtual reality," *Media Psychology*, vol. 11, no. 3, pp. 354–376, 2008.
- [3] L. J. Gerry, "Paint with me: Stimulating creativity and empathy while painting with a painter in virtual reality," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 4, pp. 1418–1426, 2017.
- [4] M. Slater and M. V. Sanchez-Vives, "Enhancing our lives with immersive virtual reality," *Frontiers in Robotics and AI*, vol. 3, p. 74, 2016.
- [5] K. Höök, *Designing with the Body: Somaesthetic Interaction Design*. MIT Press, 2018.
- [6] S. Engeser and F. Rheinberg, "Flow, performance and moderators of challenge-skill balance," *Motivation and Emotion*, vol. 32, no. 3, pp. 158–172, 2008.
- [7] A. Aron, E. N. Aron, and D. Smollan, "Inclusion of other in the self scale and the structure of interpersonal closeness," *Journal of personality and social psychology*, vol. 63, no. 4, p. 596, 1992.
- [8] O. Lartillot and P. Toivainen, "A matlab toolbox for musical feature extraction from audio," in *International conference on digital audio effects*. Bordeaux, 2007, pp. 237–244.



# Real-time Mapping of Periodic Dance Movements to Control Tempo in Electronic Dance Music

**Lilian Jap**

KTH Royal Institute of Technology  
lilianj@kth.se

**Andre Holzapfel**

KTH Royal Institute of Technology  
holzap@kth.se

## ABSTRACT

Dancing in beat to the music of one's favorite DJ leads oftentimes to a powerful and euphoric experience. In this study we investigate the effect of putting a dancer in control of music playback tempo based on a real-time estimation of body rhythm and tempo manipulation of audio. A prototype was developed and tested in collaboration with users, followed by a main study where the final prototype was evaluated. A questionnaire was provided to obtain ratings regarding subjective experience, and open-ended questions were posed in order to obtain further insights for future development. Our results imply the potential for enhanced engagement and enjoyment of the music when being able to manipulate the tempo, and document important design aspects for real-time tempo control.

## 1. INTRODUCTION

In Electronic Dance Music (EDM), a DJ combines in advance planning and real-time decisions for the purpose of creating an intense and ecstatic dance experience. Taking such dance experience into account is a direction of high potential when finding new practices for interactive systems based on the ideas of embodied interaction [1]. Current musical/technical landscapes have shifted the focus away from the passive individual towards an active role in sound [2] with the impact of embodied interaction.

Enabling the user to find an intuitive way for controlling the sound parameters of a music playback could pave the way further for an interactive musical environment. When it comes to the mapping of music playback tempo, the idea of involving dance movements for manipulation might not be a novel exploration field of research, eg. [3] and [4] that both involved techniques of video and/or image analysis. The main contribution presented in this paper is however the study of dancers' experiences when interacting with the proposed system, an autonomous interactive system that expands the experience one has while dancing with it. This might open up possibilities for individuals wishing to physically interact with music on a personal level - without the requirement of having a musical background, or the expectation of having the actual physical ability to play a musical instrument.

*Copyright: © 2019 Lilian Jap et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.*

In this paper, a proposed design system is presented where the following hypothesis is addressed: mapping of real-time measurements of a dancer's rhythmic movements to tempo manipulations in EDM can lead to a dance experience that compares positively to a standard playback of the same type of music. The definition of a dancer entails any possible user of the proposed system; whether the user possess the skill-sets of a dancer or not. A proof-of-concept prototype is presented, along with its design process and findings made after performing a user-study where participants got to interact with the prototype.

## 2. BACKGROUND

A number of studies involving experiments with music and embodiment have been conducted. In the context of musical performances, the related work appears to have a common feature; sound-producing and communicative musical gestures as an extension of the body [5]. A music performance may include gestures by those that produce sounds, and by those that perceive sounds (i.e. listeners and dancers) [6], with the larger body of research focusing on the former.

Motion sensors used with the objective of implementing different sound synthesis techniques have been implemented in various mixed interdisciplinary approaches, such as the sonification of body movements of contemporary circus artists [7], conducting [8], and music pedagogy [9]. Placing wireless motion sensors on hands and feet, it was shown that utilizing gestures as game content brings more substance to the game [10].

When it comes to hearing rhythm in music, bodily movements play an important role when developing the skill involved in rhythmic perception [11]. Audio feedback can induce more awareness as it brings deeper understanding in how the body moves [12]. Aligning rhythmic movements with rhythm in music have been used as an approach to improve the users own movement performance in the context of sports (running) as well as physical rehabilitation and health [13, 14]. The gestures of dancers, and their relation to underlying meter have been the subject of various studies employing sensor technologies [15, 16], and repetitive movements have been found to be more pronounced in hand gestures than the gestures of other body parts [15].

Marker-based infra-red motion capture (MoCap) technology provides accurate data of complex movement in a three-dimensional space [17], but is largely limited to applications in a laboratory space. Recently, Inertial Measurement Units (IMU) have been applied in movement-based inter-

action design [17–19]. Their miniature in size, mobile use and reported accurate essential values make them beneficial to use in more flexible contexts.

### 3. METHOD

In the present study, a real-time prototype was designed for estimation of periodicity in a user's body movements, and tempo manipulation of audio recordings based on these measurements. In order to establish the building blocks and parameters for the prototype, a pre-study was conducted (Section 3.1). An overview of the prototype's components is outlined in Section 3.2. The participant groups, the experimental setup, and the evaluation method based on questionnaires are described in Sections 3.3 to 3.5, respectively.

#### 3.1 Pre-study

An initial study was conducted with three participants to establish the design of the prototype. Different placements of IMU sensors on the body were examined in terms of usability as well as pronouncement in the movements (hand wrist, ankle joint, hip and the back). One sensor was used and based on the notion that the placement should serve a practical fit.

In the participants' expressed preferences, the ankle joint and hand wrist were the preferred placements as it made the sensor less noticeable or could enable more control in the tempo manipulation. However, the clearest pronunciation of measured periodic movement in the initial study was observed in the hand wrist. This corroborates findings by Leman and Naveda [15], motivating our decision to place the sensor on the hand wrist.

Adjustments in the implementation of the prototype were made as well since high latency in combination with sudden tempo changes were encountered and a confusion from the participants was expressed. The analysis frame size for estimating the periodicity of the arm movement was set to obtain a sufficiently reactive system, while still facilitating reliable periodicity estimation. Other system parameters, such as the form of tempo changes in the audio playback were also determined in this pre-study. After testing various sensor and audio processing approaches, the system design as depicted in Figure 1 emerged.

#### 3.2 Prototype

##### 3.2.1 Equipment and platform

A IMU sensor from x-io Technologies Ltd<sup>1</sup> was used, which makes use of the Open Sound Control (OSC) protocol. This opens up compatibility with other software applications, for instance Max/MSP<sup>2</sup>, which was applied to collect and process incoming data from the sensors. The real-time communication was performed via Wi-Fi using TP-Link AC750 travel router as a separate 5Ghz wireless network instead of the sensors internal antennae, allowing for future extensions using multiple sensors.

##### 3.2.2 System design

Parts of the operations were computed using JavaScript, within the Max/MSP environment. The chain of operations as depicted in Figure 1 can be described as follows:

1. OSC messages about the accelerometer data from the NGIMU sensor are received.
2. Raw accelerometer data from x-, y- and z-axis are smoothed through low-pass filtering in a Max/MSP [slide]-object, filtering with slide value  $S = 10$  according to Equation 1. A given sample output  $y_n$  is equal to the previous value  $y_{n-1}$  plus the difference between the input  $x_n$  and the previous value divided by the slide value  $S$ . Given a slide value of  $S = 1$ , the output will therefore always equal the input. Given a slide value of  $S = 10$ , the output will only change 1/10th as quickly as the input<sup>3</sup>.

$$y_n = y_{n-1} + \frac{x_n - y_{n-1}}{S} \quad (1)$$

3. The fundamental frequency  $F_0$  of the movements in the data stream in each axis is estimated by the [pipo.yin]-object from Mubu for Max-toolbox<sup>4</sup>. The object makes use of the YIN-algorithm [20], which also provide the quality factor of the detected periodicity. Several values were tested for different attributes of the yin-object in the initial study, and significant for the interaction were the sample size of frame (N), hop size (N/16), and frame-rate (sample rate of sensors). N = 100 was found to provide sufficiently accurate results while being short enough to track a speed up/speed down of the acceleration. Data streams are sliced into windowed frames of size N using the [pipo.slice]-object from the same toolbox and with the sensor's default sample send rate of 50 Hz, data streams are processed over the last 2 seconds.
4. Values of the computed quality factor - in each axis - are smoothed using a second-order moving average filter with subsets of 10 and 5 sample values, respectively.
5. The highest obtained quality factor determines which estimated  $F_0$  to be used, i.e. a choice between x-, y-, and z-axes was made based on which expresses the most consistent periodic movement.
6.  $F_0$  is converted into beats per minute (bpm).
7. Based on the changes in speed of the movements, the tempo control value for the audio playback will change accordingly. If detecting a speed increase/decrease in the dancer's movement, the current bpm value of the playback will increase/decrease with a value of 3bpm. If the movements are stopped, the playing tempo will decrease as the current design of

<sup>1</sup> <http://x-io.co.uk/ngimu/>

<sup>2</sup> <https://cycling74.com/products/max/>

<sup>3</sup> <https://docs.cycling74.com/max7/maxobject/slide>

<sup>4</sup> <http://forumnet.ircam.fr/product/mubu-en/>

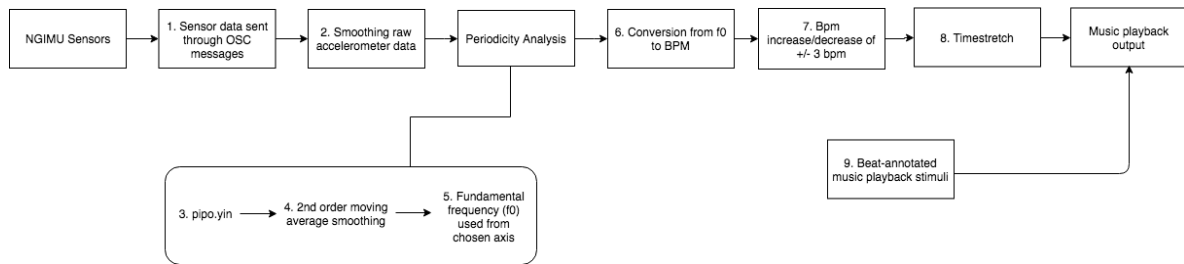


Figure 1. Block diagram of the developed prototype.

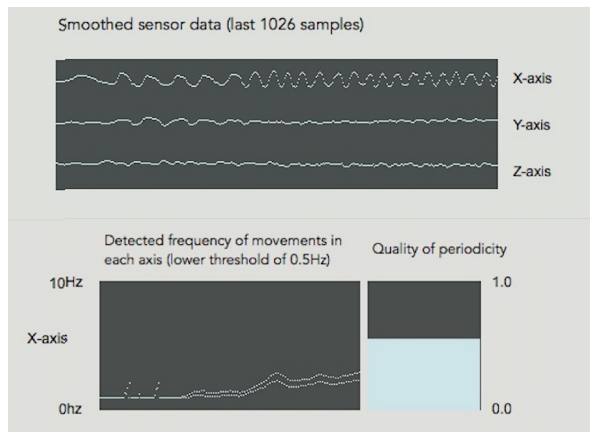


Figure 2. Screen-shots of recorded sensor data graph, detected frequency and the axis with highest estimated quality of detected periodicity. Recorded with one of the participants dancing to a EDM music sample.

the system is interpreting the movements to be in a slower state than previous data stream set. The minimum playback tempo was set to 60bpm. The initial tempo control value was set to the tempo of the original music recording.

8. Bpm-control values are sent to a sample-playback object to be time-compressed/time-stretched in real-time, based on the difference between the bpm-control value and the current playback tempo. The time compression/stretching make use of beat annotations of the played audio sample (see Block 9 in Figure 1). In future work, this may be replaced by a real-time beat estimation of the music audio signal.

Figures 2 provides an example for sensor data when a participant is moving in an intense and repetitive manner. The axis with the resulting highest estimated quality factor of detected periodicity (here, the X-axis) determines which detected frequency to be used for tempo manipulation, and the lower part of Figure 2 illustrate how a tempo increase in the oscillation on the X-axis leads to an increasing tempo estimate for the movement.

### 3.3 Participants

12 participants between the ages 22-31 participated in the study (8 men and 4 women, mean age 27 years). 9 par-



Figure 3. Two of the participants testing the prototype in the main study.

ticipants have a background in dance or are working professionally with dance. All participants were reported to be in a healthy condition. Each participant was recorded individually and written consent was obtained before the experiment started.

### 3.4 Experimental Setup

The participants were offered to choose music samples themselves. However, since no participant preferred this option, music stimuli were randomly chosen for each session within a range of 110-140 BPM from a collection of 33 recent EDM productions<sup>5</sup>. The duration of each session was kept within 15 minutes as a way to keep the participant engaged.

In one session, two experiments were performed for each participant. In the first experiment no tempo manipulations were conducted, and the participant was instructed "to move freely, but repetitively to the presented music stimuli". The second experiment included the same task but the ability to control the tempo of the music stimuli through the implemented prototype. The NGIMU was in both sessions placed on the right hand wrist of the participant. The experiments were conducted in a personal living room using 2.0 stereo speakers with Bluetooth for playback of music. The participants were also video recorded to facilitate further analysis of spontaneous reactions and interactions. Figure 3 shows screen-shots from two participants' recordings.

### 3.5 Questionnaires

After each session, the participants were asked to fill a questionnaire<sup>6</sup>, which contained both open- and closed-

<sup>5</sup> The list of songs is provided here: <https://bit.ly/2DhrLO9>

<sup>6</sup> The questionnaire, including all responses, can be obtained from <https://bit.ly/2WGJ0RU>

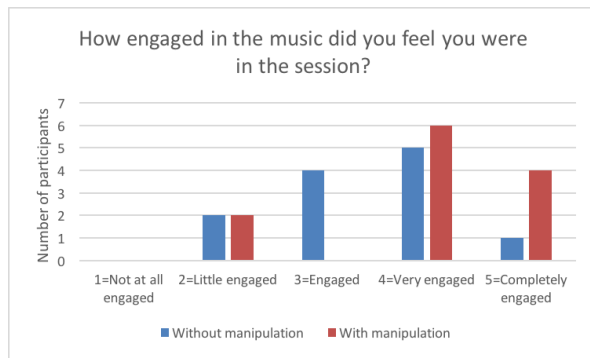


Figure 4. Results of participant ratings of how *engaged* in the music they were in the first (no tempo manipulation) and second (with tempo manipulation) session.

ended questions to gather qualitative and quantitative data. The questionnaire focused on enabling comparison of the participants experience to standard playback to an interactive setting. In addition, several questions were provided to gather more qualitative suggestions regarding the system design.

## 4. RESULTS

### 4.1 Engagement in music

Figure 4 shows the participants' ratings when asked about how engaged in the music they were, comparing first (no tempo manipulation) and second session (with tempo manipulation). The x-axis represents the distribution of the participants rating, where 1 = not at all engaged, 2 = little engaged, 3 = moderately engaged, 4 = very engaged and 5 = completely engaged. The y-axis represents the number of participants. The first session without tempo manipulation gave mean value rating of 3.42, while the second session with the ability to manipulate tempo gave a mean value rating of 4.00. Two participants gave a lower rating in the second session (both from 3 to 2), five provided the same ratings, and another five increased ratings.

Participants who had given a lower rating (2 - little engaged) would give the motivation that dancing alone in a room made them feeling little engaged as to whenever he/she would be in a club-like situation: "I think it was not easy to be dancing alone."(P1), "I was all alone. Dancing is kind of a social experience for me."(P8)

Participant (P8) gave this first session a rating of 2 (little engaged) while the second session got a rating of 5 (completely engaged). For the reason that:

I can't say that my dancing improved, but it was really engaging when you could control the tempo with your movements. (P8)

Another participant who gave a different rating the second session (value of 4 – very engaged), in comparison to the first session (value of 2 – little engaged) explained the difference as:

Because I was in control of the music. It made it more of a "game" than just dancing to music. (P6)

In relation to the first session, where the motivation for the rating was explained as:

Hard to lose yourself in the music when you are alone in room like this. You feel watched even though the room is empty. It's easier to dance when you're in a room full of people dancing, or at home, where you feel completely relaxed. (P6)

Another participant who had given a higher rating the first session (value 3 – moderately engaged) but had a different engagement the second session (value 2 – little engaged) explained it as:

The changing tempo made it hard for me to enjoy the music and dance to it. As I moved to the pace of the music it somehow did not catch my movement and began to slow down which made me have to wave my arm fast to make the music speed up again (...) On the other hand, I felt like I got to interact with the music in a new way. The ability to adjust the music as if I was dj-ing was cool, as I could play with it. The songs also sounded cool when switching the tempo. (P3)

Other participants who had shifted from feeling moderately engaged to either very or completely engaged gave the following reasons:

The possibility to change the dynamic through my movement was for me more exciting. As well as no need to stay repetitive. (P4)

Because my moves and actions had an impact on the source/reason why I was originally moving. It created a little bubble in which a conversation with myself could happen. (P5)

### 4.2 Enjoyment when dancing

Figure 5 shows questionnaire responses when the participants were asked to compare enjoyment of dancing between the first and second session. In addition to comparing the engagement ratings for the individual sessions (previous subsection), these ratings provide an additional comparison from the perspective of the participant. The x-axis represents the rating from 1 (=much worse) to 5 (=much better). The y-axis represents the number of participants giving a certain rating. Following results gave a mean value rating of 3.08.

When asked about how they felt being able to modify the tempo, a majority described a positive feeling. But along with a positive feeling, some still expressed having a somewhat split feeling about the interaction.



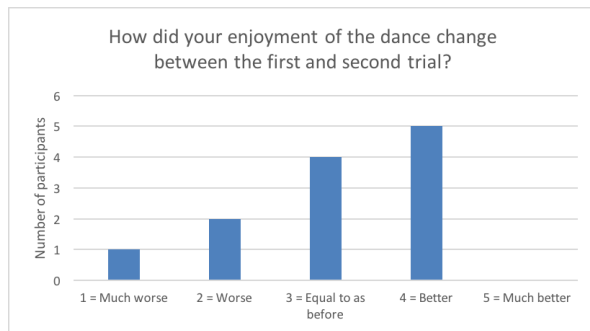


Figure 5. The rating when comparing the participants *enjoyment* of dancing between the first and second session.

It was playful, an enjoyable negotiation. (P4)

It was interesting and fun. At the same time there was a feeling of responsibility towards the tempo in comparison to the music that is not tempo modified. Like if it would be my mistake if the dance floor died. (P2)

It felt interesting but exhausting to keep up as my arm had to stay in one tempo even though my body might have wanted to working in contradiction to the music sometimes. But it was also interesting to hear when I got tired and then realizing that I had physically changed tempo. (P11)

Great but confusing at times. It made me move in a certain way to be sure to not mess up the tempo. Felt a bit restricted. (P12)

A common feeling of restriction as the last-mentioned could be identified among other participants as well.

It was hard to use it, as there was a delay of a few seconds, and as I normally adjust to the beat and have a difficult time to set the pace for it to play, as I then need to move faster than the music. (P3)

It slowed down to easily in my opinion. I often felt that the tempo was perfect, but it always slowed down a few seconds later. (P6)

(...) I felt more engaged in one way, because I could control the pace of the music. Although I felt more restricted because I had to more repetitive and less instinctual. (P9)

A couple participants described a feeling of uncertainty in if he/she is doing right.

(...) you realise that you sense of beat has gotten worse. (P7)

(...) perhaps a slight misconception from my part when the vocals kicked in as I didnt feel that I had as much control over them. (P10)

### 4.3 Spontaneous reactions

Recognized among the participants was how a larger part of the participants generated more arm movements during their second session in comparison to their first session. Moving the arm as an indication of exploring possibilities in the tempo manipulation, going from one extreme to another (fast/slow, periodic/non-periodic) was often followed by reacting with a laughter. There were furthermore participants who appeared to shift between adjusting their dance to what they were hearing and interfere their dance by generating arm gestures to control the tempo.<sup>7</sup>

## 5. DISCUSSION

In this study, rhythmic movements of dancers were analyzed in real-time for their predominant periodicity, which was then mapped to manipulate tempo in the music playback the dancer was moving to. The emphasis in the user-study and resulting evaluation was put on the participants' subjective experience, both with and without the ability to control the tempo of the music. Even though no statistical significance emerges from our study, the results indicate the potential of positive dance experiences when improving the system based on the comments of our study.

Preserving the sound quality of the input audio was one of the main challenges. Granted that the prototype aimed to strictly change the bpm of the playing input audio – without affecting other sound parameters – the music's characteristics and the sound quality of it were still affected. A possible explanation for how most of the participants felt more engagement and/or enjoyment could be grounded in a feeling that it is music they themselves somewhat created. Thus, an enhanced feeling in their engagement and/or enjoyment in the interaction. Likewise, positive effect on engagement could be originating from the fact that they had to execute more control and needed to be more attentive to the details in the music playback.

Modifications that became significant for the user interaction were found to be situated in the functionality on how the playback tempo changes. The developed system applies small but noticeable changes within a short time span for the user to sense the agency in the interaction.

Among the participants' expressed opinions, the most common criticism was regarding the delay between a change in body movement and a tempo change in the playback. In the way that the current prototype is constructed, the detected frequency is always analyzed comparing current subset to previous subset. If the frequency is analyzed to be higher, the tempo will increase. If lower, the tempo will decrease. Thus, attempting to make a tempo change for a short time period might result in a playback tempo manipulation. It therefore requires the user to create faster movements for a longer time period in order to make movement changes noticeable in sound.

After having arrived at a desired tempo of the playback, the frequency and clarity of repetitive movements was fre-

<sup>7</sup> Video examples of users in the tempo-manipulation experiment: <https://youtu.be/3toOXtS2bKI>, <https://youtu.be/i63UBMehWGs>, <https://youtu.be/livNDDOkxeQ>.

quently observed to decrease. This made P3 feel that the system did not catch the pace of the movements and therefore this participant pointed out decreased enjoyment of the dance in the interactive setup.

As a potential drawback of this study, a majority of the participants were educated or are professional dancers. For participants who have a deep background in dance performance, it is assumed that taking on the task given in this study can happen effortlessly and possibly feel more engaged to any music that is played for them. Having a larger as well as more diverse population, the distribution of feedback given from the participants may well have differed. However, involving experienced dancers enabled us to get a rich body of verbalizations that can guide further development.

### 5.1 Future research

Some of the participants expressed confusion about which sound parameters in the musical structure were controlled by body movement, even though it was limited to tempo alone. Further study on what other manipulations based on gestures can therefore be explored.

Studying how a beat synchronization would influence the interaction if it were to be done more “musically” can be of interest, *e.g.* changing the tempo only at the beginnings of bars. This can provide a solution for the user to feel more engaged in the dance if wanting to break off from the interaction and stay in the tempo. It is also likely to improve the interaction experience by detecting durations during which a clear signal is not received from the motion data, and deactivating the tempo manipulation in such phases.

In order for the system to be adaptive to as many gestural vocabularies as possible, additional features to the interaction can be considered as a way to give the user further choice and/or control in his or her movements. This allows the user to reconstruct the system as desired to accommodate his or her gestural preferences and/or capabilities, as was the case in Mulder’s work of GRIP instruments [21].

Investigating how participants would interact in groups is worth exploring, and could add dimensions of social interaction through entrainment to the interaction.

## 6. CONCLUSION

The initial objective of this study was to investigate the subjective experience when users are given control of the decision-making in the music that is played for them. A proof-of-concept prototype was built and examined by a total of 12 participants. A user-study was conducted consisting of two sessions, one without tempo manipulations by the prototype and one with tempo manipulations controlled by periodic body movement. The proposed design is suggested to provide a dance experience that can compare positively to a standard playback of EDM music. Results imply giving an overall positive dance experience worth exploring further. For a number of the participants, the prototype indicated contributing to more engagement and enjoyment than to a standard playback of EDM in-

volving not interacting with the prototype. The qualitative statements provide a rich set of directions to develop the prototype towards increased robustness and diversity of interactions.

## 7. REFERENCES

- [1] P. Dourish, *Where the action is*. MIT press Cambridge, 2001.
- [2] A. Tanaka, “Music one participates in,” in *Proceedings of the 8th ACM conference on Creativity and cognition*. ACM, 2011, pp. 105–106.
- [3] G. Castellano, R. Bresin, A. Camurri, and G. Volpe, “User-centered control of audio and visual expressive feedback by full-body movements,” in *Affective Computing and Intelligent Interaction*, 2007, pp. 501–510.
- [4] C. Guedes, “Controlling musical tempo from dance movement in real-time: A possible approach,” in *Proceedings from the International Computer Music Conference*. ICMC, 2007, pp. 453–457.
- [5] S. Dahl and A. Friberg, “Visual perception of expressiveness in musicians’ body movements,” *Music Perception: An Interdisciplinary Journal*, vol. 24, no. 5, pp. 433–454, 2007.
- [6] A. R. Jensenius and M. M. Wanderley, “Musical gestures: Concepts and methods in research,” in *Musical Gestures*. Routledge, 2010, pp. 24–47.
- [7] L. Elblaus, M. Goïna, M.-A. Robitaille, and R. Bresin, “Modes of sonic interaction in circus: Three proofs of concept,” in *Sound and Music Computing Conference*, 2014.
- [8] P.-J. Maes, D. Amelynck, M. Lesaffre, M. Leman, and D. Arvind, “The conducting master: an interactive, real-time gesture monitoring system based on spatiotemporal motion templates,” *International Journal of Human-Computer Interaction*, vol. 29, no. 7, pp. 471–487, 2013.
- [9] F. Bevilacqua, F. Gu  dy, N. Schnell, E. Fl  ty, and N. Leroy, “Wireless sensor interface and gesture-follower for music pedagogy,” in *Proceedings of the 7th international conference on New interfaces for musical expression*. ACM, 2007, pp. 124–129.
- [10] Y. Jung and B. Cha, “Gesture recognition based on motion inertial sensors for ubiquitous interactive game contents,” *IETE Technical review*, vol. 27, no. 2, pp. 158–166, 2010.
- [11] B. M. Costello, *Rhythm, Play and Interaction Design*. Springer, 2018.
- [12] Q. Wang, P. Markopoulos, B. Yu, W. Chen, and A. Timmermans, “Interactive wearable systems for upper body rehabilitation: a systematic review,” *Journal of neuroengineering and rehabilitation*, vol. 14, no. 1, p. 20, 2017.

- [13] J. Hockman, M. M. Wanderley, and I. Fujinaga, “Real-time phase vocoder manipulation by runner’s pace.” in *NIME*. Citeseer, 2009, pp. 90–93.
- [14] B. Moens, C. Muller, L. van Noorden, M. Franěk, B. Celie, J. Boone, J. Bourgois, and M. Leman, “Encouraging spontaneous synchronisation with d-jogger, an adaptive music player that aligns movement and music,” *PloS one*, vol. 9, no. 12, p. e114234, 2014.
- [15] M. Leman and L. Naveda, “Basic gestures as spatiotemporal reference frames for repetitive dance/music patterns in samba and charleston,” *Music Perception: An Interdisciplinary Journal*, vol. 28, no. 1, pp. 71–91, 2010.
- [16] P. Toiviainen, G. Luck, and M. R. Thompson, “Embodied meter: Hierarchical eigenmodes in music-induced movement,” *Music Perception: An Interdisciplinary Journal*, vol. 28, no. 1, pp. 59–70, 2010.
- [17] R. T. Solberg and A. R. Jensenius, “Optical or inertial? evaluation of two motion capture systems for studies of dancing to electronic dance music,” in *Sound and Music Computing Conference (SMC)*, 2016.
- [18] R. Niewiadomski, M. Mancini, S. Piana, P. Alborno, G. Volpe, and A. Camurri, “Low-intrusive recognition of expressive movement qualities,” in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*. ACM, 2017, pp. 230–237.
- [19] F. Visi, E. Coorevits, R. Schramm, and E. R. Miranda, “Musical instruments, body movement, space, and motion data: Music as an emergent multimodal choreography,” *Human Technology*, vol. 13, no. 1, 2017.
- [20] A. de Cheveigné and H. Kawahara, “YIN, a fundamental frequency estimator for speech and music,” *Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.
- [21] A. Mulder, “Getting a grip on alternate controllers: Addressing the variability of gestural expression in musical instrument design,” *Leonardo Music Journal*, vol. 6, pp. 33–40, 1996.

# INCREASING ACCESS TO MUSIC IN SEN SETTINGS

**Tom Davis**  
Bournemouth University  
tdavis@bournemouth.ac.uk

**Daniel Pierson**  
Bournemouth University  
contact@danpiersonaudio.com

**Ann Bevan**  
Bournemouth University  
abevan@bournemouth.ac.uk

## ABSTRACT

This paper presents some of the outcomes of a one year Higher Education Innovation Fund<sup>1</sup> funded project examining the use of music technology to increase access to music for children within special educational need (SEN) settings. Despite the widely acknowledged benefits of interacting with music for children with SEN there are a number of well documented barriers to access [1, 2, 3]. These barriers take a number of forms including financial, knowledge based or attitudinal. The aims of this project were to assess the current music technology provision in SEN schools within a particular part of the Dorset region, UK, determine the barriers they were facing and develop strategies to help the schools overcome these barriers. An overriding concern for this project was to leave the schools with lasting benefit and meaningful change. As such an Action Research [4] methodology was followed, which has at its heart an understanding of the participants as co-researchers helping ensure any solutions presented met the needs of the stakeholders. The presumption by the researchers was that the schools needed new technology to help overcome barriers. However, although technological solutions to problems were presented to the school, it was found that the main issues were around the flexibility of equipment to be used in different locations, staff time and staff attitudes to technology. These issues were addressed through the Action Research methodology to ensure that the technology designed worked for these particular use case scenarios.

## 1. INTRODUCTION

There have been several major reviews of music technology's use within SEN settings; within a general SEN educational context [1, 2, 3, 5], as well as particularly from a music therapy perspective [6, 7, 8, 9]. This growing body of literature supports the view that there is a growing interest in the use and the study of the use of music technology (MT) within these environments and by communities of practitioners. Music is used within SEN settings to support a range of activities, for example, formal class room based music teaching, one-on-one music therapy sessions, group music sessions [1, 3, 8] as well as being embedded in 'everyday' class room activities such as, signposting when it is time to get ready for lunch, or when to put your shoes on [3].

<sup>1</sup> HIEF funding is allocated by Research England with a remit to to support and develop a broad range of knowledge-based

### 1.1 Benefits of music

Music has been identified as having a number of benefits in terms of promoting health and wellbeing, as well as having the ability to develop wider skills relating to participation, socialisation, attention and fine motor skills [1] [3]. The employment of MT has a long history of being utilised (for example, see [10, 11]) to help provide access to music making, particularly for those working within SEN settings where bespoke technologies can be used to overcome some of the physical or cognitive barriers that may be present for these children in using 'standard' acoustic instruments [6]. Recently there has been an increase in the amount of bespoke music technologies, specifically designed for the SEN sector, (for example [12, 13, 14, 15]) as well as a proliferation of music delivered through tablet based devices such as Apple's iPad [16, 17, 18]. (See Ward et. al. [5] for a full review).

### 1.2 Issues of access

In 2011, the UK Charity, Youth Music commissioned a review of engagement with MT in special educational and disabled music settings throughout the UK. This report [1] clearly sets out the benefits of using music technology for SEN children and young people, but also identified many barriers to the use of MT within these settings. In this review, these barriers are summarised under three headings: A need for specialist training; Resources; and A fear and dislike or indifference to technology [1 p. 31]. I will borrow these categorisations to revisit these issues.

#### 1.2.1 Need for Specialist training

As already mentioned, there are a wide range of technologies available to enable the delivery of music within a classroom, ranging from bespoke hardware controllers to software running on tablet devices. These devices require a level of specialist knowledge to be useable and to interface with existing equipment. Like all technology, these devices are prone to constant change and upgrade cycles, and as such, there is a continual need for specialist training. In addition to the Youth Music report [1] this need for training has also been identified by a range of other authors: Welch et. al. [2] recognised a lack of knowledge and understanding of music technology among music therapists and teachers within a SEN setting. Of the 80% of sampled schools that used distance sensing technology in music, only 11% used them on a weekly basis. A UK based

interactions between universities and the wider world, which result in benefits to the economy and society.



survey of practice and attitudes to electronic technologies in clinical music therapy carried out by Magee [6] found that 65% of music therapists felt they had a lack of skills in using this type of equipment. This translated into the electronic technology equipment being ‘in a box in a cupboard’ [6 p.144]. A more recent study in 2012, by Hahna, Hadley, Miller and Bonaventura [19], surveyed 600 music therapists from the US, Australia, Canada and the UK. This found that 61% of respondents were self-taught, suggesting that more training was needed to make ‘more technology accessible to a variety of learners’ [19 p.456].

### 1.2.2 Lack of Resources

Lack of resources can be material in nature, for example, lack of physical technology; lack of funds to purchase technology; or it could be more intangible in nature. For example, lack of information about how to integrate the technology into the sessions; or simply lack of time to utilise the technology or learn about the technology. Findings from an international survey of music therapy practitioners by Hadley et. al. [20] reports on the barriers to entry as ‘lack of money, lack of professional experience, lack of portability, lack of time to learn, limits of the facility, lack of interest, a belief that music technology is not appropriate to music therapy clinical work, or that music technology was not appropriate for their particular clientele’ [20]. Farrimond et. al [1] identify barriers to MT provision around the area of cost of technology. They draw on Nagler [21] who found that the ‘high cost of new equipment’ was a barrier and Magee [6], who states that 40% of respondents to her survey of Music Therapists identify MT as being ‘too expensive to buy’ [6]. In more recent publications (2017) cost seems to be less of an issue, with a recent focus on tablet based interfaces [9, 3] suggesting that the relative affordability of tablet based applications for MT is increasing provision. Welch et. al. [3] suggest that 79% of schools have access to music through apps on tablets, with 65% having access to music software such as Garage band [16, 3, p. 9]. Despite the proliferation of tablet devices in schools and the availability of low cost or free apps, it is worth noting that tablet based activities are not suitable for all children, access to technology will vary with need, and bespoke technologies for MT can still be prohibitively expensive. (For example a new Soundbeam 6 [14] is around £2,500). The proliferation of available MT itself can become a barrier as Knight and Krout [9] note, the challenge that the sheer number of resources itself provides a challenge for the music therapist, in terms of knowing and evaluating which approach is best for their client [9].

### 1.2.3 Fear, dislike or indifference of technology

A fear, dislike or indifference of technology is Farrimond et. al.’s [1] 3<sup>rd</sup> category. This is supported by statistics from Magee [6], showing that 18% of therapists stated that they did not like technology and 4% thought that music technology was not appropriate/relevant for the clients they were working with [6 p. 143]. The most recent PROMSIE report [3] does note some marked improvements in the sector in the use and uptake of music compared with the similar survey of 2011 [2]: ‘with more musically qualified

staffing, a broader range of resources for the music curriculum, more external organisations available to support music, increased use of music technology and improved music therapy provision’ [3, p.3]. However, the report does not specifically identify the attitude of staff towards the technology but it seems hopeful that with increased availability of technology within schools that Farrimond et al.’s [1] prediction has come true that the ‘apparent acceptance of conventional technology might positively influence any negative perceptions of music technology over time’ [1, p.33]. Despite all the positive outcomes from the PROMSIE report there is no data on the actual use of MT within schools, in fact the report states ‘[o]ne caveat to these details is the extent to which, notwithstanding availability, schools regularly use such devices. Some comments suggested that this was not always the case [3, p. 9]. This is a sentiment echoed by others, for example Hadley et. al. state ‘[d]espite the passage of time, these barriers are still the same as quoted in Magee (2006)’ [20]. Despite the seven years passing since Farrimond et. al.’s [1] review and the increase in the availability of MT based solutions, there remains many barriers to entry to using these technologies in the classroom and within music therapy contexts. Issues seem to still be present regarding ease of use, cost (for specific specialist equipment), and especially around training of how to operate and how to integrate technology into the class room environment.

## 2. ACTION RESEARCH METHODOLOGY

Action Research is presented in Reason and Bradbury’s SAGE handbook of Action Research [4] as following the following working definition: ‘Action research is a participatory process concerned with developing practical knowing in the pursuit of worthwhile human purposes. It seeks to bring together action and reflection, theory and practice, in participation with others, in the pursuit of practical solution to issues of pressing concern to people, and more general the flourishing of individual persons and their communities’ [4, p. 4]. In contrast to conducting research on subjects as objects, Action Research is very much conducted with stakeholders as “co-researchers” [4, p. 9] and has a primary purpose to “produce practical knowledge that is useful to people in the everyday conduct of their lives” [4, p. 4].

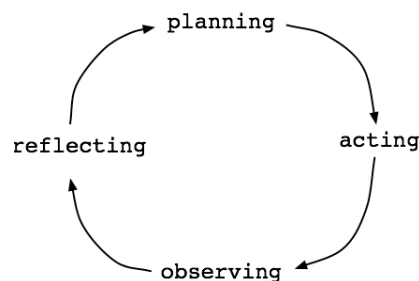


Figure 1. Action-reflection cycle [24, p.57]

An Action Research based methodology is iterative in nature with most projects following a cyclical process of action and reflection based on Lewin’s [22] theory of action as a spiral of steps involving planning, fact-finding

and execution. Action Research's more recent models, such as that outlined by McNiff [23], describe a cycle of Planning, acting, observing, reflecting, planning.. etc. [23, p. 57].

Action Research was considered a suitable methodological approach for this project as its tenet of affecting change within a community aligns well with the ethos of this project to empower and leave a lasting beneficial change in communities. The spirit of working with participants as co-researchers also ties into ethical concerns relating to 'expert researchers' telling practitioners what to do. This is a particular concern when working with communities of marginalised voices such as the disabled. Since another aim of this project was to empower existing stakeholders and create a community of practice, it was very important that the stakeholders felt part of the research process and that their opinions were valued at least as highly as the researchers themselves. The Action Research cyclical process of observing, reflecting, planning etc. mirrors those found in participant design and as such helps ensure that any results from the project meet the needs of all the stakeholders involved.

### 3. CASE STUDY – SCHOOL A

Access to local schools was facilitated through Coda. Coda is a local charity who states their objectives on their website as: 'Coda uses music as a tool for personal and social change. We love music and believe in its power to transform lives. Coda provides a place to learn, train and develop, and we offer help and support through participation and therapy' [24]. Coda were already facilitating some training based around the use of music technology within local schools. As an entry point to the project we were invited to give a short presentation about the project and its goals to representatives from a number of local SEN schools that had attended one of these training sessions. After this meeting an open call to partake in the project was sent out to all attendees at this session and two schools responded. The scope of this article is around the outcomes of working with one of these schools.

#### 3.1 Data Gathering

Data was primarily gathered through minutes of stakeholder meetings, open or semi-structured interviews and reflective writing by the researchers. Stakeholder meetings were initially with AA, Pupil Technology and AAC lead - & BB, Head of lower school (ex music teacher), Tom Davis (TD), Lead Researcher and Daniel Pierson (DP) Research Assistant. Later meetings were generally with AA, TD & DP and a range of pupils from the school. There was also a project steering group consisting of TD, DP, Ann Bevan (AB) and Phil Hallet (PH) from Coda.

#### 3.2 The School

School A provides education for pupils who have severe, profound and multiple learning difficulties. The large majority of pupils have one or more additional needs, including autistic spectrum disorders, medical needs, sensory impairments and emotional, social and mental health

difficulties. Ofsted school inspection report 2015. School A describe themselves in their literature as 'a specialist school for children and young people who have Complex Learning Difficulties or Disabilities' They have a wide range of pupils with a range of needs, but quite a large proportion with Profound Multiple Learning Difficulties, with an associated wide range of varied and complex needs.

#### 3.3 Current Music Provision

It was evident from the first meeting with the stakeholders that there was a passion for music and a great desire to include more of it within the school. BB stated that up to recently (Academic Year 2015-16), they had had a dedicated music therapist, but due to funding constraints this was stopped. Also, "years ago" (no timescale given) they used to have a dedicated music teacher that retired and not replaced. Instead a decision was made that music would be delivered by the class teachers. Music is used in a lot of ways in the ordinary classes, from helping with routine through to teachers delivering music lessons. There was some feeling that the music provision by 'normal' class teachers was difficult as they may not be trained specialists in music. Music is also used to make everyday teaching more accessible. For example, if reading a story there are audio cues to engage the pupils and music is used in a lot of ways to facilitate their learning. Music is also used throughout the day to help structure activities, such as a song for putting on their shoes, getting ready for lunch etc. School A has some outside support in delivering music. Coda, a music charity comes into the school and runs weekly sessions with the pupils. This normally culminates in a performance or a project.

#### 3.4 Main barriers to access

##### 3.4.1 Staff Perspective

Issues at School A still echo those outlined by Farrimond et al [1]: staff not musically trained; issues around asking non-musically trained staff to come up with a deliver music based activities and a fear and dislike of technology. For example, comments suggesting that the staff were scared of technology. Staff comments about the Soundbeam 2, a commonly used piece of equipment in the school; 'it's technology', 'it's big', 'there's a lot wires'; "Oh man, it's got more than 3 buttons" AA. An additional issue raised was the issue of tight time constraints for both teachers and support workers. 'It can't take something like 30 minutes to setup it needs to be plug and play ... People struggle with time ... So I think the impact is that it really has to be something that can be used by everybody in the school and that every member of staff should be able to use without too much help' AA – Interview July 2017.

##### 3.4.2 Student Perspective

The students at School A have a range of learning difficulties often with combinations of issues. As identified at the first meeting the main barriers to access from student perspective were: difficulty physically accessing things; issues of motor control and lack of grip strength. Another issue identified was visual impairment (VI), not as a standalone condition but paired with other learning

difficulties and disabilities. Two pieces of equipment that were particularly identified as being preferred by the pupils were the resonance board (a sheet of wood slightly raised off the ground designed to resonate and amplify acoustic sound), and the OmiVista [25] (an interactive floor projection system).

#### 3.4.3 Outcomes of first stakeholder meeting

As the majority of researchers on the project had a music technology background there was a tendency for the team to propose technical solutions to the perceived problems. An idea to come from the first meeting was that the researchers' thought that the school would benefit from the design and construction of an active vibro-tactile resonance board that could be linked to the OmiVista to make it more interactive. This idea was proposed to the stakeholders at the 2<sup>nd</sup> meeting. Issues that arose from this meeting were that the board needed to be easily accessible and easy to use. The school has a current vibro-tactile resonant board, but it is located in a sensory room and any sound has to go through a specific Hi-Fi. This presented a number of problems around accessibility. There is only one sensory room in the school and students are allocated time in there in relation to need. This means that not all pupils get access to this space. The board in there is also quite high off the ground which means that students have to be hoisted into position on the board. This takes time and some students are dependent on their wheelchairs and can't be hoisted. The board can only be used to play sounds through and isn't an interactive environment for the pupils to take part in. 'The challenges for the boards are that they need to be portable – i.e. so you can take them into as many lessons as possible. They need to be easy to use. They need to just plug them in and they work. They need to give good vibro-tactile feedback to the students – the students need to want to use them...' (TD Reflective writing.)

### 3.5 Resonance Board Development



**Figure 2.** Reckhorn BS-200 Body Shaker [26] mounted on small resonance board

Following the requirements outlined in the stakeholder meetings and subsequent interviews the team created a vibro-tactile resonance board that could be taken anywhere in the school.

This vibro-tactile resonance board consisted of a Reckhorn BS-200 Body Shaker [26], a low frequency transducer such as those used in gaming chairs, connected to a plywood board. The transducer was driven by

a 100W amplifier and was positioned to try and create an even distribution of frequencies across the board. The transducer outputs as low as 5Hz but also produces vibrations in the audio range, meaning that you get a tactile as well as an audible output.



**Figure 3.** The larger resonance board.

Initially a small board was made (610mm by 1220 mm by 18mm). The board was raised off the floor with some pine runners and the transducer mounted underneath. Importantly this board was low enough and strong enough to support an electric wheelchair. This meant that if needed, students could access the board without needing to be removed from the wheelchairs. The tactile vibrations are strong enough to be felt through the chair, albeit in a reduced manner. The board went through a number of evaluation sessions with AA and a number of different pupils. A larger board was also created, (1220mm by 2240mm by 18mm) which was designed for larger/older pupils to lie down on. The larger board was also painted white to enable easy projection of the OmiVista [25] onto its surface. Also the amplifier on both models was swapped for a less powerful model [27] that was smaller and could be attached to the underside of the board. This meant that the board could be used just by plugging in anything that has a 3.5mm audio jack output and operated with one power switch.



**Figure 4.** Pupil using the resonance board with a Skoog [12].

### 3.6 Evaluation of the Resonance Board

Since most of the students that used the system could not verbalise, we relied very much on their careers' assessment of their level of engagement. However, as you can see from the examples below generally there was very positive feedback. So much so that students were foregoing their normal preferred instrument of choice, i.e. the guitar



in favour of the new system. The following two transcripts are just some of the examples that demonstrate evidence that the students had valuable interactions with the resonance board.

### 3.6.1 Example 1

Transcript with Pupil 1, AA and a Teaching Assistant (TA) – DP also in attendance. (5<sup>th</sup> May 2017).

TA: Oh Hello , what is going on?

AA: He is absolutely loving it.

TA: You've got the piano?

AA: Yes but it is much more than the normal piano because he has got the sensation as well.

TA: Clever people. That's amazing.

AA: He was lying really, really still and then moving.

TA: Amazing!

AA: It's awesome isn't it.

TA: It really is.

### 3.6.2 Example 2

Pupil 2 using large resonance board wired up to the Omi-Vista. AA & DP in attendance 5th May 2017. AA then took a guitar from the shelf and put it near the pupil. He strummed the guitar only briefly before rolling away from the guitar onto the other side of the board. AA was surprised by this, saying "Wow, that's quite telling if the guitar doesn't get any attention!" This was a key moment as AA recalled it even after the session was over, saying "I'm amazed, because he always would go for the guitar". She explained that "If somebody walks into the room with a guitar he's like this-" motioning outstretched arms towards the guitar. AA: "I'm amazed because he will always go for the guitar and he just didn't. No, not interested in that thank you" (5th May video 2017).

## 3.7 Legacy of the project

### 3.7.1 Lasting change.

The funding period finished in July 2017 and the equipment was left with the school without any further follow up or support. Researchers returned to the school in January 2018, 7 months after the end of the project to see if there had been any lasting changes in the school. On arriving at the school, AA took TD up to the classroom to show the resonance board in use. In the interview that followed it transpired that the boards are being put to ongoing and continuous use. They have been used with a variety of existing equipment within the school including, the Beamz [13], the Skoog [12], microphones, and iPad apps. The school have gone as far as actually purchasing an additional Skoog [12] so that they have an extra one to use with the new board. The only continued barrier to access was with using it with the OmiVista [25]. The OmiVista itself still needed modification to work with board. (A side panel needed unscrewing to access the audio output). This was a barrier for the staff, and a health and safety concern for AA so the board was not being used in this way.

### 3.7.2 Use in the classroom

On asking if the board had increased access to music for these students AH replied: "The context hasn't always

been in the context of making music – but certainly they have been experiencing sound in a different way" (AA 19<sup>th</sup> Jan. 2018). In general the board has increased access to experiencing sound for the children. The existing resonance board is too tricky to use. It is high up such that wheel chairs can't be put on it. The sound system in there is too complicated and there isn't as much vibration from the box itself. In contrast this solution 'everybody can use it. If you can plug in a pair of headphones you can use it.' AH 19<sup>th</sup> Jan 2018) Having a moveable board means that more children get access as it can be used not only in specific music lessons (these happen in the room with the large resonance board in which is not so portable) but rather in the standard classrooms. This means it can be used not just for the delivery of music but anytime that they use music/sound throughout the day, which adds enrichment to all sorts of activities.

### 3.7.3 Results of training

AA had recently run a training session with the board as part of an inset day to a large group of staff. Mostly, at the moment, staff are trying ideas suggested by AA, but there's a lot of interest and people are excited about using it. As AA states: 'I think the other thing is that after the training, people are more excited about it. Which means that if people are excited about something they want to do it. So it means that music happens. Where as before it might not have happened much. I think that that is a big difference actually. That people want to do it.' AA.

### 3.7.4 Impact of small interventions

I would like to share one event that demonstrates the amount of impact such a small change in providing access can have on a child and their careers experience in school. One child, with hearing impairment normally does not react in any way to sound. He cannot leave his wheel chair so cannot use the current resonance board setup in the sensory room.

"one of the teachers came up to me and said, we had an all tears moment. because one student doesn't normally react at all to music. And it was a really, really big reaction. but they were like, we were all crying. (Laughing) That's really, really, lovely. (AA, interview 19<sup>th</sup> Jan 2018) I've just seen photographs of the board session that made all the staff go all teary and I have to say, it nearly got me too!!! To see the reaction of someone with massive sensory impairment feeling the rhythm coming up through his wheelchair really is awesome. Unfortunately I cannot release any pictures of this specific student, but I can tell you that it put a massive lump in my throat. (AA email correspondence 25<sup>th</sup> Jan 2018)

## 4. FINAL THOUGHTS

This project shows that despite improvements documented in the literature, many SEN schools still have issues accessing music through technology. The main barriers are not to do with the technology itself but, rather, with their context of use. Difficulties arise with either a lack of knowledge of how to use the technology, either from a technological perspective (how do I turn it on?) or from a



musical perspective (how do I use this technology to deliver music?). This project again highlights the real need to create outputs that work with and for stakeholders. The Action Research methodology helped assure that any design decisions benefited the stakeholders and ultimately made the finished technologies useable within the school context. Although the assumption going into this project was that the solution would be in the development of new technologies, the technologies developed in this project are not new, in fact they are really modifications of technologies that were already found in the school. What is different, is the ease of which they can be used in a variety of different environments. The flexibility to just move them to different classrooms, to plug them into a range of input devices (depending on pupils needs) and the ability to use them with pupils whilst in wheelchairs meant that pupils who normally didn't have access, suddenly had access to music. This ease of use, and associated staff training, meant that staff were willing to try the technology, so ultimately it was integrated into everyday school activities. Most gratifyingly, you can see from the final correspondence from the school, what impact these small changes can have on an individual's life experience.

**Acknowledgments-** HEIF funding, the schools, staff and children who for ethical reasons can't be named.

## 5. REFERENCES

- [1] B. Farrimond, D. Gillard, D. Bott. & D. Lonie, "Engagement with Technology in Special Educational and Disabled Music Settings," Youth Music Report, [Online], 2011. Available: <https://network.youthmusic.org.uk/file/5694/download?token=I-1K0qhQ>
- [2] G. Welch, A. Ockelford, & S. Zimmermann, "Provision of Music in Special Education (PROMISE)", London: Royal National Institute for the Blind (RNIB)/Institute of Education, University of London, 2011.
- [3] G. Welch, A. Ockelford, & S. Zimmermann, E. Wilde, "The Provision of Music in Special Education (PROMISE) 2015". Presented at the 26th International Seminar of the ISME Commission on Research, London 18- 22 July 2016. Proceedings, pp 292–303.
- [4] P. Reason, & H. Bradbury, H. The Sage Handbook of Action Research: Participative Inquiry and Practice. Second Edition. SAGE Publications Ltd. London, 2008.
- [5] A. Ward, T. Davis, & A. Bevan. "Music Technology and Alternate Controllers for Clients with Complex Needs," *Music Therapy Perspectives*. To appear 2019
- [6] W. Magee, "Electronic Technologies in Clinical Music Therapy: A Survey of Practice and Attitudes," *Technology & Disability*, 18(3), pp 139-146, 2006.
- [7] W. Magee, Music technology in therapeutic health settings. London: Jessica Kingsley. 2014.
- [8] B. J. Crowe, & R. Rio, "Implications of technology in music therapy practice and research for music therapy education: a review of literature," *Journal of Music Therapy*, 41(4), 282–320, 2004.
- [9] A. Knight, & R. E. Krout, "Making Sense of Today's Electronic Music Technology Resources for Music Therapy," *Music Therapy Perspectives*, 35(2) pp 219-225, 2017.
- [10] J. C. Nagler, & M.H. Lee, "Use of Microcomputers in the Music Therapy Process of a Postviral Encephalitic Musician," *Medical Problems of Performing Artists*, 2(2), pp 72-74, 1987.
- [11] E. Krout, "Integrating Technology", *Music Therapy Perspectives* 8(1), pp 8-9, 1990.
- [12] Skoog Available: <http://skoogmusic.com>
- [13] Beamz Available: <https://thebeamz.com>
- [14] Soundbeam Available: <https://www.soundbeam.co.uk>
- [15] Musii Available: <https://musii.co.uk/>
- [16] GarageBand Available: <https://www.apple.com/uk/mac/garageband/>
- [17] ThumbJam Available: <https://thumbjam.com>
- [18] Clarion, Open Up Music. Available: <http://openupmusic.org/the-clarion/>
- [19] N. D. Hahna, S. Hadley, V. H. Miller, & M. Bonaventura, "Music technology usage in music therapy: A survey of practice," *Arts in Psychotherapy*, 39(5), pp 456–464, 2012.
- [20] S. Hadley, N. D. Hahna, V. H. Miller, & M. Bonaventura, "Setting the Scene: An Overview of the Use of Music Technology in Practice," in *Music Technology in Therapeutic and Health Settings*. Magee, W. Ed. London and Philadelphia, Jessica Kingsley. 2014. pp 25-43.
- [21] J. C. Nagler, "A Qualitative Study of Children in Crisis: Interventions Through Music Therapy and Digital Music Technology", Ph.D. dissertation., New York University, 1993.
- [22] K. Lewin. "Action research and minority problems," *Journal of Social Issues*, 2(4): pp 34–46, 1946.
- [23] J. McNiff. "Action Research: Principles and Practice," 3rd Edition. Oxon, Routledge, 2013.
- [24] CODA Available: <http://coda.org.uk>
- [25] OmiVista Available: <http://omi.uk/omivista-interactive-floor/>
- [26] Reckhorn BS-200 Available: <https://www.reckhorn.net/pages/body-shaker.php>
- [27] Topping VX1 Available: <http://en.tpdz.net>

# INTERACTING WITH MUSEBOTS (THAT DON'T REALLY LISTEN)

Arne Eigenfeldt

School for the Contemporary Arts,  
Simon Fraser University  
arne\_e@sfu.ca

## ABSTRACT

*tinySounds* is a collaborative work for live performer and musebot ensemble. Musebots are autonomous musical agents that interact, via messaging, to create a musical performance with or without human interaction.

## 1. INTRODUCTION

Generative and interactive systems have a long history within music [1, 2, 3]; more recently, aspects of artificial intelligence have been applied to such systems, creating a contemporary approach known as metacreation [4]. One useful model borrowed from artificial intelligence is that of *agents*, specifically multi-agent systems. Agents have been defined as autonomous, social, reactive and proactive [5], similar attributes required of performers in improvisation ensembles. Musebots [6] offer a structure for the design of *musical agents*, allowing for a communal compositional approach [7] as well as a unified model. An overview of recent musebot ensembles is given elsewhere [8].

## 2. MUSEBOTS

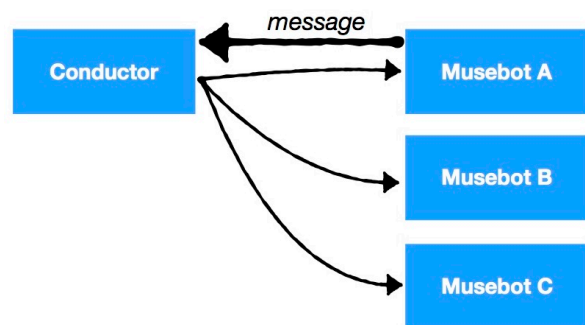
Musebots are pieces of software that autonomously create music collaboratively with other musebots. They decide how to respond to their environment – and each other – on their own, based upon their internal beliefs, desires, and intentions.

The musebot protocol<sup>1</sup> is, at its heart, a method of communicating *states* and *intentions*, sending networked messages established through a collaborative document via OSC [9]. A *Conductor* serves as a running time generator, as well as a hub through which all messages pass (see Fig.1).

Copyright: © 2019 Arne Eigenfeldt. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> <http://tinyurl.com/gngmews>

Individual musebots broadcast to the ensemble aspects of their performance; the details of *what* they communicate is left to the designer of the ensemble.



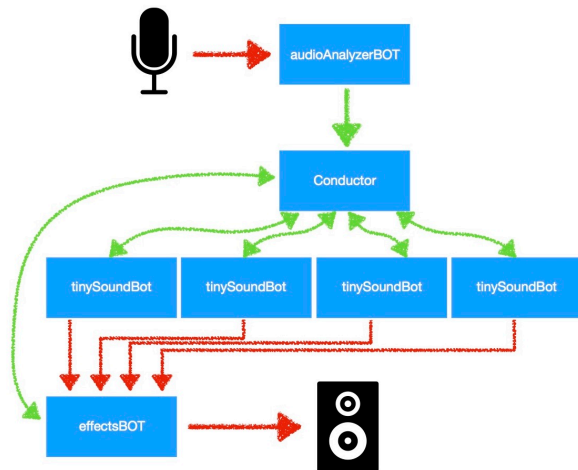
**Figure 1.** Diagram of messages between musebots and the Conductor. In this case, Musebot A sends a broadcast message to the Conductor, who rebroadcasts it to the ensemble.

## 3. TINYSOUNDS: FOR VOICE AND MUSEBOT ENSEMBLE

The musebot ensemble in *tinySounds* is a redeployment of an earlier metacreative system, *The Indifference Engine*, which is partially described elsewhere [10]. Live audio is analyzed for features: spectral centroid; spectral flux; loudness; activity level (onset detection); and Bark band spectrum. This information is messaged to the audio musebots and an effectsBot (see Fig.2). This latter musebot adds effects – delay, pitch shift, time stretch, ring modulation, and distortion – autonomously, based upon its interpretation of the analysis messages. For example, it will switch effects when activity is low, and add more processing when flux is high.

The audio musebots – in this case, four instances of *tinySoundBot* – have access to a large corpus of pre-analyzed soundfiles; given a Bark band spectral analysis via the *Conductor*, the audioBots will attempt to find the closest matching recordings from their available database. The audioBots autonomously begin and end playing based upon incoming messages, including activity and flux, as well as reacting to whether other audioBots are active or not.

Audio is generated using a modified version of *CataRT* [11].



**Figure 2.** Diagram of musebots in *tinySounds*. Audio paths are in red; musebot messages are in green.

#### 4. PERFORMANCE NOTE

Machine learning algorithms are wonderful for sifting through data and discovering relationships; more challenging is how these algorithms can be used for generation. It isn't that difficult, for example, to train a system to provide similar sounds for a database, given a live sound. But what's the artistic interest in that? Similarly, it isn't that difficult to extract live performance information from an improvising musician – activity level, general frequency range, timbre – so that the system responds likewise. But, again, reactive systems lose interest fairly quickly.

I find it much more interesting when my musebots go off on their own, exploring their own ideas through beliefs they may have formed incorrectly and unintentionally. For that reason, I usually build a lot of ambiguity into my analysis or provide conflicting information. What happens when one musebot is sure of something, while another is absolutely sure of something else? And what if a third musebot just doesn't care?

In *tinySounds*, musebots are trained using a neural net on a corpus that has been hand-tagged for valence and arousal measures, as well as pre-analysed for spectral information. However, the correlation between audio features (what the musebots are listening for) and affect (valence and arousal) isn't direct; in assigning the latter, I may decide that a sound from the corpus is complex and active, but my reasons for doing so may not use the same information as the musebots are provided with. Thus, a musebot may decide that, based upon what it has learned, a live sound is high valence / high arousal, but the listener may perceive it otherwise. This isn't a flaw in the system; it's a feature!

Lastly, my role as overseer in the musebot ensemble allows me to further disrupt how the musebots apply their knowledge. The corpus is organized semantically (i.e. voice sounds, kitchen sounds, transportation sounds, etc.); once a musebot is using a certain subdirectory, it can't easily switch to another. As a result, its choice of related sound, whether affective or timbral, is limited to

what is immediately available to it. If the musebots are frustrated, they haven't mentioned it to me (yet).

Musebots are not straightforward reactive processes; instead, they have their own beliefs (in this case, the incoming analysis data), desires, and intentions. They will happily play on their own, or they may react very closely to the live performance; more often than not, they will offer their own "reinterpretation" of the live performance, with individual reactions to the analysis data.

#### Acknowledgments

This research was made possible through a Social Sciences and Humanities Research Council of Canada (SSHRC) Insight Grant.

#### 5. REFERENCES

- [1] J. Chadabe, "Interactive music performance system," U.S. Patent No. 4,716,804. 5 Jan. 1988.
- [2] R. Rowe, *Interactive music systems: machine listening and composing*. MIT press, 1992.
- [3] T. Winkler, *Composing Interactive Music: Techniques and Ideas Using Max*. MIT press, 2001.
- [4] M. Whitelaw, *Metacreation: Art and artificial life*. Cambridge MA: The MIT Press, 2004.
- [5] M. Wooldridge and N. Jennings, "Intelligent Agents: Theory and Practice," in *Knowledge Engineering Review* 10/2, 1995.
- [6] O. Bown, B. Carey and A. Eigenfeldt, "Manifesto for a Musebot Ensemble: A Platform for Live Interactive Performance Between Multiple Autonomous Musical Agents," in *Proceedings of the 21st International Symposium of Electronic Arts, Vancouver*, 2015.
- [7] A. Eigenfeldt, O. Bown and B. Carey, "Collaborative Composition with Creative Systems: Reflections on the First Musebot Ensemble," in *Proceedings of the 6th International Conference on Computational Creativity, Park City*, 2015.
- [8] A. Eigenfeldt, "Musebots: Collaborative Composition with Creative Systems," in *eContact! 20.2 TIES 2017: 11th Toronto International Electroacoustic Symposium, Toronto*, 2018.
- [9] M. Wright, "Open Sound Control: A New Protocol for Communicating with Sound Synthesizers," in *Proceedings of the 23rd International Computer Music Conference, Thessaloniki*, 1997.
- [10] A. Eigenfeldt, "Generating Structure – Towards Large-scale Formal Generation," in *Proceedings of the Tenth Artificial Intelligence and Interactive Digital Entertainment Conference, Raleigh*, 2014.
- [11] D. Schwarz, "Corpus-based Concatenative Synthesis," in *IEEE Signal Processing Magazine*, 24(2), 2007.

# EXTENDING JAMSKETCH: AN IMPROVISATION SUPPORT SYSTEM

Akane Yasuhara, Junko Fujii, and Tetsuro Kitahara

College of Humanities and Sciences, Nihon University, Japan

yasuhara@kthrlab.jp, fujii@kthrlab.jp, kitahara@kthrlab.jp

## ABSTRACT

We previously introduced JamSketch, a system which enabled users to improvise music by drawing a melodic outline. However, users could not control the rhythm and intensity of the generated melody. Here, we present extensions to JamSketch to enable rhythm and intensity control.

## 1. INTRODUCTION

Improvisation is an enjoyable but difficult form of music performance because musicians must create melodies while playing an instrument. Therefore, to enable non-musicians to improvise easily, various systems have been proposed [1–5]. For example, JamSketch [5] enables users to play an improvisation by drawing a curve called a *melodic outline*, which represents the overall shape of a melody, with a mouse or their finger on a piano-roll display. This approach does not require skill in playing an instrument, but the melodic outline is limited in its expressivity because it only represents how the melody moves up and down in pitch. Until now, users could not control the rhythm or intensity of the melody.

In this paper, we extend JamSketch to enable users to control the rhythm and intensity of the generated melody when drawing a melodic outline. Our intention is to add functionality while not making the system more complex or less intuitive to use. To satisfy these requirements, we adopt the following approaches:



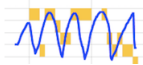
**Rhythm:** We support only the control of note density (how many notes appear within one bar) to keep the operation simple. Users can control the note density by changing the waviness of the melodic outline.

**Intensity:** Through the use of devices supporting pen pressure sensing (e.g., Microsoft Surface Pro), the system allows users to control the intensity by changing the pen pressure.

## 2. SYSTEM

Once the system launches, the piano-roll display with its horizontal time axis and vertical pitch axis appears. The user draws a melodic outline with a stylus pen supporting

Table 1. Waviness and note density

$0 \leq s \leq 10$	no-wave		$D = 6$
$10 < s \leq 50$	small-wave		$D = 12$
$s > 50$	large-wave		$D = 2$

pen pressure sensing. The shape of the melodic outline is reflected in the pitch of the generated melody, and the pen pressure is reflected in the intensity of the performed melody. Also, the rhythm (note density) is controlled by drawing wavy curves.

### 2.1 Drawing a melodic outline

On the piano-roll display, the user draws a melodic outline. The melodic outline is displayed, and the line weight represents the pen pressure when the outline is drawn.

### 2.2 Analyzing the waviness of the melodic outline

Once the user draws a melodic outline, the system analyzes the waviness of the outline separately for each bar. Let  $y(t)$  ( $t = 1, \dots, T$ ) be the melodic outline at a certain bar (the resolution of  $t$  depends on the screen's resolution). The system calculates a smoothed outline  $\bar{y}(t) = \frac{1}{\tau_0} \sum_{\tau=0}^{\tau_0-1} y(t+\tau)$ .

Then,  $\delta(t) = \bar{y}(t) - y(t)$  is calculated. After that,

$$s = \sqrt{\frac{1}{T-1} \sum_{t=1}^{T-1} (\delta(t+1) - \delta(t))^2}$$

is calculated. The waviness at the corresponding bar is determined as follows (Table 1):

$$\text{waviness} = \begin{cases} \text{"no-wave"} & (0 \leq s \leq 10) \\ \text{"small-wave"} & (10 < s \leq 50) \\ \text{"large-wave"} & (s > 50) \end{cases}$$

### 2.3 Determining the rhythm

The rhythm is determined separately for each bar and is represented as a binary vector where 1 stands for an onset and 0 stands for a non-onset. Because the shortest duration is set to an eighth-note triplet in the current implementation, the rhythm at each bar is represented as a 12-dimensional binary vector  $R = (r_0, \dots, r_{11})$  ( $r_i \in \{0, 1\}$ ).

Copyright: © 2019 Akane Yasuhara et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



First, a tentative rhythm  $R' = (r'_0, \dots, r'_{11})$  is generated from the melodic outline. The basic policy is to generate a note onset at time points when the melodic outline has a high gradient. Let  $y'(i)$  be a down-sampled melodic outline (the time resolution is an eighth-note triplet). Then,

$$r'_i = \begin{cases} 1 & (|y'(i) - y'(i-1)| > \epsilon) \\ 0 & (\text{otherwise}) \end{cases}$$

is calculated, where  $\epsilon$  is a threshold.

Then, the rhythm  $R$  is determined with a genetic algorithm (GA) with the fitness function defined as follows:

$$F(R) = w_0 \text{sim}(R) + w_1 \text{lik}(R) + w_2 \text{dens}(R)$$

where

- $\text{sim}(R)$  is the similarity to the tentative rhythm  $R'$ :

$$\text{sim}(R) = - \sum_{i=0}^{11} (r_i - r'_i)^2.$$

- $\text{lik}(R)$  represents the musical likelihood of  $R$ :

$$\text{lik}(R) = \sum_{i=0}^{11} \log P(r_i|i),$$

where  $P(r_i|i)$  is the conditional probability of  $r_i$  given the time index  $i$  and is calculated from a dataset.

- $\text{dens}(R)$  represents how well the melody's rhythm follows the waviness of the melodic outline:

$$\text{dens}(R) = - \left( D - \sum_{i=0}^{11} r_i \right)^2,$$

where  $D$  is a preferred note density determined from the waviness of the melodic outline. In the current implementation,  $D$  is set according to Table 1.

## 2.4 Determining the pitch

After the rhythm of the melody is determined, the pitch (note number) of each note is determined. This is also based on a genetic algorithm with a fitness function that tries to maximize both the melody's closeness to the melodic outline and its musical likelihood calculated from a melody dataset. See [5] for details.

## 2.5 Determining the velocity



The velocity of each note is determined according to the pen pressure at the corresponding point in the melodic outline. The mapping between the pen pressure and the velocity in the current implementation is listed in Table 2.

## 3. EXAMPLES

An example of melody generation including different note densities is shown in Fig. 1 (left). From the 2nd to 5th measures, the melodic outline contains a small wave, and accordingly, the generated melody contains many short notes. From the 8th to 12th measures, the generated melody consists of a fewer longer notes because the melodic outline contains a large wave.

An example of controlling the velocity is shown in Fig. 1 (right). From the 4th to 5th measures, the velocity is high because the pen pressure is high (so, the curve is thick).

Table 2. Pen pressure and velocity

Pen pressure	Display	Velocity
3000 or higher		127
2000 to 3000		80
1000 to 2000		50
lower than 1000		30

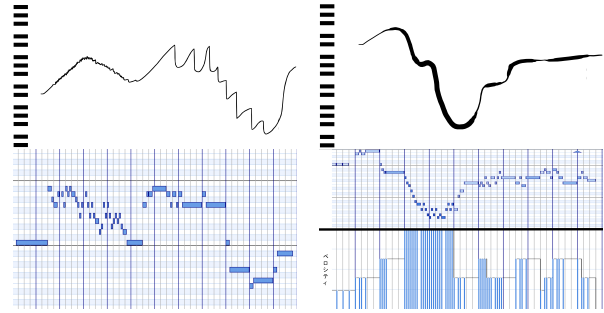


Figure 1. Examples of melodic outlines (upper) and generated melodies (lower) with rhythm (left) and intensity (right) control. The piano-roll representation includes the velocity data at the right-side figure.

## 4. CONCLUSION

In this paper, we presented two extensions (rhythm and intensity control) of the JamSketch system, which supports improvisation by non-musicians. We conducted experiments to confirm the effectiveness of this system. We omitted the results due to a lack of space, but we will report them in a separate paper.

**Acknowledgments:** This project is supported by JSPS Kakenhi JP16K16180, JP16H01744, JP16KT0136, and JP17H00749 as well as the Kawai Foundation for Sound Technology and Music.

## 5. REFERENCES

- [1] S. Fels, K. Nishimoto, and K. Mase, "MusiKalscope: A graphical musical instrument," *IEEE Multimedia*, vol. 5, no. 3, pp. 26–35, 1998.
- [2] H. Miyashita and K. Nishimoto, "Theremoscore: A new-type musical score with temperature sensation," in *Int'l Conf. New Interface for Musical Expression*, 2004, pp. 104–107.
- [3] K. Ishida, T. Kitahara, and M. Takeda, "ism: Improvisation supporting system based on melody correction," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, June 2004, pp. 177–180.
- [4] J. Buchholz, E. Lee, J. Klein, and J. Borchers, "co-JIVE: a system to support collaborative jazz improvisation," Aachener Informatik-Berichte RWTH Aachen, Department of Computer Science, <http://www.informatik.rwth-aachen.de/go/id/loj/lidx/1/file/47944>, Tech. Rep. AIB-2007-04, 2007.
- [5] T. Kitahara, S. Giraldo, and R. Ramírez, "Jamsketch: Improvisation support system with ga-based melody creation from user's drawing," in *Proc. of Int'l Symp. on Computer Music Multidisciplinary Research*, 2017, pp. 352–363.

# VISUALIZING MUSIC GENRES USING A TOPIC MODEL

**Swaroop Panda**

IIT Kanpur  
pandas@iitk.ac.in

**Vinay P. Namboodiri**

IIT Kanpur  
vinaypn@iitk.ac.in

**Shatarupa Thakurta Roy**

IIT Kanpur  
stroy@iitk.ac.in

## ABSTRACT

Music Genres serve as an important meta-data in the field of music information retrieval and have been widely used for music classification and analysis tasks. Visualizing these music genres can thus be helpful for music exploration, archival and recommendation. Probabilistic topic models have been very successful in modelling text documents. In this work, we visualize music genres using a probabilistic topic model. Unlike text documents, audio is continuous and needs to be sliced into smaller segments. We use simple MFCC features of these segments as musical words. We apply the topic model on the corpus and subsequently use the genre annotations of the data to interpret and visualize the latent space.

## 1. INTRODUCTION

Music genre visualizations have not caught enough attention. Probabilistic Topic Models [1], have found wide applications in the field of Natural Language processing. We use an unsupervised topic model on music genres data for visualization. For our work we use raw music files, in .wav format. Unlike text documents, raw music data has no discrete components such as words. To create a text-like corpus, we slice the audio data into smaller segments. We use MFCC features of these smaller slices as the representation. Further, to build a corpus, we create a feature dictionary by using the k-means algorithm. Also, in text documents, the inferred topics form a collection of words and hence are straightforward to interpret. In our case, musical words, which are mere MFCC feature arrays, lack inherent meaning and cannot be interpreted. We interpret the latent space of the topic model using genre annotations in the dataset.

## 2. RELATED WORK

[2] had discussed some audio visualization techniques in MIR which are mostly signal processing based. Topic Models have also been applied on audio [3] for the purpose of audio information retrieval. We use the fault-filtered GTZAN [4] dataset for genre analysis which is popular in MIR community.

Copyright: © 2019 Swaroop Panda et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 3. PROBABILISTIC TOPIC MODEL

Probabilistic topic models are based upon an idea that documents are mixture of topics and topics are probability distributions over words. These words come from a fixed size vocabulary. To make a new document, one chooses a distribution of topics, then chooses a topic from this distribution and finally draws a word from the chosen topic. The Latent Dirichlet Allocation (LDA) inverts this generative process and thus infers the set of topics that were that useful in generating the document. The plate notation below defines the generative process.

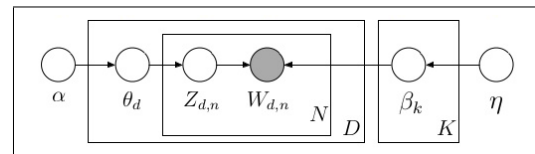


Figure 1. Plate Notation for a Topic Model

## 4. APPLYING TOPIC MODELS ON MUSIC

The main challenge with the application of topic models in the music (with raw audio files) is to represent the audio in a text-document like corpus. The intent of the work is to interpret the latent space of the topic model using music genres. This interpretation would help giving a probabilistic genre annotation to a song. For example a song may belong 60 % to Blues, 15 % to Jazz and 25 % to Pop genres. To enable such a probabilistic assignments, we build basic genre buckets consisting of at least 3 genres. We do this since a mixture containing all the 10 genres would be very large and obfuscating for the listener to meaningfully interpret. The rationale used to bucket the genres is roughly based on the histories and the musical form of these genres. The first bucket consists of Rock, Metal and Pop genres; the second of Blues, Jazz and Country genres and the final bucket consists of Reggae, Disco and Hip-Hop genres. The songs were clipped down to 0.10 seconds clips. The MFCC features of these clips were then calculated. We then use a K means clustering algorithm on the MFCC features to build the dictionary. It partitions the data into k clusters, where each data point belongs to a cluster and the cluster mean serves as its prototype.

## 5. INTERPRETATION OF THE LATENT SPACE

Unlike text documents, where topics are interpreted as a mixture of words; the acoustic topic model has topics which

are mixtures of cluster means. These cluster means are prototypes of the nearest datapoints (the audiofiles) and thus lack meaning. It is thus essential to assign a suitable meaning to these cluster means. The first part of the interpretation involves understanding the cluster means in terms of music genre. The cluster means are constructed from the MFCC arrays. The cluster means hence can be mapped to and from these audio files and linked with the genre annotations. For instance, let's say that 3 audio files, audio1, audio4 and audio7 make up a cluster mean. We get back to the dataset and find out that audio1 belongs to the Blues genre, audio4 belongs to the Country genre and the audio7 belongs to the Blues genre. Hence, the genres associated with cluster means becomes Blues, Country, Blues. In math, the cluster centers (or terms) can be described as the following,

$$\begin{aligned} clustermean1 &= \{Blues, Country, Blues\} \\ &= 0.67Blues + 0.33Country \\ clustermean2 &= \{Blues, Jazz, Jazz, Country\} \\ &= 0.25Blues + 0.5Jazz + 0.25Country \end{aligned} \quad (1)$$

Once we interpret cluster means in terms of music genres, we can conveniently represent the topics in terms of music genres. The topic space consists of cluster means and an associated probability value. The cluster means can now be defined as music genres with their proportions.

$$\begin{aligned} Topic1 &= prob1 * clustermean1 + prob2 * clustermean2 \\ &= (prob1 * 0.67 + prob2 * 0.25)Blues + \\ &\quad (prob1 * 0.33 + prob2 * 0.25)Country \\ &\quad + (prob2 * 0.5)Jazz \end{aligned} \quad (2)$$

Once the topic space has been interpreted, the document-topic proportions can also be made sense of. The document topic proportions from the topic model are probability values of the inferred topics present in each document. In this context, the document topic proportions can provide with the proportions of different music genres present within the musical document, that is, a song. The term topic proportions are proportions of different topics for a term in the document. These term topic proportions can be similarly interpreted in terms of genre proportions.

$$Doc1 = prob1 * topic1 + prob2 * topic2 \quad (3)$$

## 6. EVALUATING THE TOPIC MODEL

We evaluate the model using a genre classification task. We use the model to get the document-topic proportions of every document (song). We use these document-topic proportions as a representation for each song. We use genre labels from the fault-filtered GTZAN dataset, divide the data into train-test sets and perform the classification task using a SVM. We also test our model with different number of topics to look for the optimal number of topics that best capture the genre bucket.

	2	3	4	5
1	0.47	0.53	0.58	0.53
2	0.36	0.38	0.35	0.40
3	0.53	0.46	0.48	0.50

Table 1. Accuracies obtained for different number of topic terms. The rows represent the genre bucket, while the columns represent the number of topics

## 7. MUSIC GENRE VISUALIZATION

Using the topic model, we can get a probabilistic genre labels of different songs (from document-topic proportions) along with progressive genre visualizations (from term-topic proportions).



Figure 2. Probabilistic Genre Labels

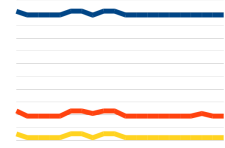


Figure 3. Progressive Genre Visualization

Every song has a probabilistic genre annotation which can be visualized by the doughnut chart in Figure 2; where each colour represents a music genre. Similarly, any song can be represented by a progressive genre visualization, where the y-axis represents time, the x-axis genre proportion and the colours respective music genres.

## 8. FUTURE WORK

The dataset is too small for the Topic Model to be very effective. Also, the topic model works on the bag-of-words assumption; which may not be too efficient for modelling music data. Other effective music representation techniques can be used. Moreover, different kinds of Topic models can be tuned to work specifically for music data.

## 9. REFERENCES

- [1] D. M. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.
- [2] M. Cooper, J. Foote, E. Pampalk, and G. Tzanetakis, "Visualization in audio-based music information retrieval," *Computer Music Journal*, vol. 30, no. 2, pp. 42–62, 2006.
- [3] S. Kim, P. Georgiou, and S. Narayanan, "Latent acoustic topic models for unstructured audio classification," *APSIPA Transactions on Signal and Information Processing*, vol. 1, 2012.
- [4] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

# CompoVOX: REAL-TIME SONIFICATION OF VOICE

**Daniel Molina Villota**

University of Málaga, Spain  
Jean Monnet University, France  
danielmolinavillota@gmail.com

**Isabel Barbancho, Antonio Jurado Navas**

University of Málaga, Spain  
ibp@ic.uma.es, navas@ic.uma.es

## ABSTRACT

It has been developed an interactive application that allows sonify human voice and visualize a graphic interface in relation to the sounds produced. This program has been developed in MAX MSP, and it takes the spoken voice signal, and from its treatment, it allows to generate an automatic and tonal musical composition.

## 1. INTRODUCTION

The main objective of any interactive application, is to strongly move the viewer to play with that application and to show him novel aspects. For that reason, using voice as a generator of tonal musical sequences, can be interesting both for composers and amateur users, since they can explore aspects of the voice that are not usually threatened for sound generation.

Voice is the most important tool of human communication, but also the most important musical instrument, therefore, the non-daily use of voice (whether communicative or musical) based on a synesthetic treatment as well as not synesthetic but with both at same time can be quite interactive.

CompoVOX has been developed in MAX MSP using the tools of data processing, generation of visual effects and sound effects offered by this programming method.

## 2. TALKING ABOUT INTERACTIVITY

In the last two decades, there has been a great technological advance in processing capacity, therefore use of technologies applicable to the numerical arts has changed very much. Here they are mentioned some interesting works related to this work.

The importance of interactivity (directly or indirectly) can be seen in projects like Opto-isolator [1], who induces the viewer to a high interaction, it's proposed that it would be the work itself who observes the viewer. Otherwise there are project called Re: MARK, who employs voice to create image employing voice's analysis (identification of phonemes), and the movement of the participants to produce real-time animations. The importance of an interface who allows viewer to play actively is very important. projects

*Copyright: © 2019 D. Molina Villota et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.*

like WIP [4] show the relevance of this fact. WIP is a project who takes curve and the amplitude of sound to allow in real time generate multiple visual combinations, projection of the geometric shapes and appearance modification.

This work has seemed inspired by the fact of being very attractive visually, that is achievable using real time synesthetic and no synesthetic procedures to generate sound and image, and at same time creating a tonal music sequence from voice and taking to generate visual forms related to the sound performed on the screen shown in the installation.

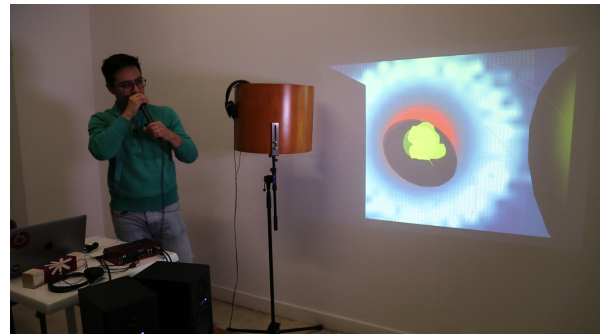


Figure 1. Doing a demo of CompoVOX

## 3. SOUND SYNTHESIS

To carry out the generation of sound, this project has made use of the acquisition of voice through a dynamic microphone. This microphone allows to isolate the environment and in this way, it is possible to focus only on the capture of voice, also an audio card is used, then the signal is captured by the microphone and transferred to the computer through the audio card.

This project has been developed through a program built under the structure of MAX MSP, the program is responsible for taking the audio signal and through different stages of treatment, obtain a signal that can be translated into midi values and for so in a tonal musical sequence.

At first, this program uses the fast Fourier transform, in this way, the region of interest for this project is filtered, that is the spectrum of the human voice that ranges between 50 and 600 Hz. Once this process has been carried out different Fourier filters are performed in parallel, which focus on taking only the signal of certain regions of the spectrum. To define such regions of interest, several tests have been made by differentiating the levels reached



in each region and that level is responsible for activating different areas of the musical scale, usually the low notes activate lower sounds and the high notes will activate higher sounds.

It is evident that the levels of each one of the signals that are obtained will vary rapidly, since they come directly from the voice signal, therefore to smooth the changes that are in the signal an averaging function is used, this averaging avoids sudden variations in the control signals, thus eliminating intrusions of sound from the environment and at the same time a signal that varies more slowly facilitates the control in real time of musical parameters that must be audible to the user.

The entire system is controlled by the same clock, but each synthesis stage is activated only when an appropriate level range is reached in the indicated region of the spectrum and in that case, there will be a mapping (by means of a scaling between frequency and level) of midi notes, all sound generation systems are passed through a tonal filter, which forces the system to have some regularity and stay in the same key.



Figure 2. Student playing with device

The different parts of the signal are used both to control the type of note and to control the attack and the temporal length of the note. A control parameter is assigned to each signal. On the other hand, sound synthesis is also done by taking the central frequency of the voice and its variations to generate a sequence of notes that is passed through a tonal filter, and subsequently the frequency value serves as a tone control parameter of a synthesizer.

For the graphic interface, several aspects are used. A background treated by MAX MSP that is responsible for reproducing a space environment and whose edges come from noise, a conical gang that covers the space and that varies in sharpness and size proportionally to the central frequency of the voice, a flat circular object located in the center it takes different forms correspondingly to the waveform of the voice, and objects that change shape according to the level in a region of low frequencies of the spectrum. The whole system moves proportionally to the level of sound, that is, the louder the voice, the closer the objects are, and the fainter they are, the farther they go. The frequency is also responsible for changing the color of the objects.

#### 4. PERFORMING THIS DEMO

During the presentation, the sounds that the person makes with the microphone will be captured. In real time, the participant's voice will be used to filter and give different colors to the sampled sounds. The voice will also be used as a mean of controlling and generating patterns / sequences that will be reproduced by the sequencer. This filtering and sequence will generate in a synesthetic manner a visual analogue that will show the details of the spectrum of the participant's voice. Additionally, the participant will be able to play with a rhythmic base.

#### 5. CONCLUSIONS

Sonification of voice signal for the generation of tonal musical sequences accompanied by a graphic interface that relates some aspects of the voice with sounds and images, is highly interactive and allows exploring aspects of sound generation that are not traditionally used in the voice treatment. CompoVox allows to work voice signal in a different way from traditional use in music.

#### 6. REFERENCES

- [1] G. Levin, "OptoIsolator". Project at M.I.T Massachusetts technological Institute. 2007.
- [2] Z. Lieberman, G. Levin, "Re: MARK". Project for FutureLab of Ars Electronics (Siemens). 2002.
- [3] N. Nordmann, J.R.Valentin, "Variationen für Tenorion". Project presented at Kunstvolkslauf Hannover, Zinnober in 2013.
- [4] B.Guesnon, "W.I.P Work.In.Processing". Project by himself begun at 2013

# FACIAL ACTIVITY DETECTION TO MONITOR ATTENTION AND FATIGUE

Óscar Cobos, Jorge Munilla, Ana M. Barbancho, Isabel Barbancho, Lorenzo J. Tardón

Universidad de Málaga, Andalucía Tech, E.T.S.I. Telecomunicación,

Dpt. Ingeniería de Comunicaciones, Campus Teatinos s/n, 29071 Málaga

ocm@ic.uma.es, munilla@ic.uma.es, abp@ic.uma.es, ibp@ic.uma.es, lorenzo@ic.uma.es

## ABSTRACT

In this contribution, we present a facial activity detection system using image processing and machine learning techniques. Facial activity detection allows monitoring people emotional states, attention, fatigue, reactions to different situations, etc., in a non-intrusive way. The designed system can be used in many fields such as education and musical perception. Monitoring the facial activity of a person can help us to know if it is necessary to take a break, change the type of music that is being listened to or modify the way of teaching the class.

## 1. INTRODUCTION

Human-machine interaction systems have improved with facial recognition. Pioneering works emerged between the 70s and 80s [1], when Facial Action Coding System (FACS) was developed. Facial activity detection allows monitoring people emotional states, attention, fatigue, reactions to different situations, etc., in a non-intrusive way. These detection can be used in many fields such as education and musical perception.

In this contribution we present a multi-purpose facial activity detection system to monitor attention and fatigue. The system has been made entirely in Matlab, making use of two specialized toolbox: Computer Vision System and USB Webcam. The operation of the system needs a computer and a Webcam USB2.0 with a frame rate of at least 60 fps.

## 2. SYSTEM DESCRIPTION

The general structure of the facial activity detection system developed in this work to monitor attention and fatigue is shown in Fig. 1. In this figure, it can be seen that the developed system consists on different stages that, although arranged in a row, are strongly related to each other with feedback information. The first state is intended to detect a face in each frame of the video as well as the Regions Of Interest (ROIs) on it: eyes and mouth. Then, the ROIs are tracked, so that even if the person moves, the regions of interest are always located in the scene. Once they ROIs are determined in any frame, it must be decided the status of each face part: eyes open or close and mouth open

or close. Further, the movement and head turns are determined. Finally, a temporary analysis of the results is made to determine drowsiness using the frequency and duration of blinking and yawning.

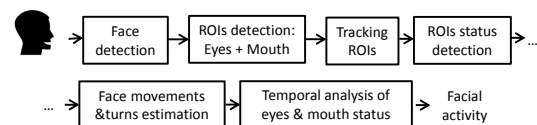


Figure 1. General structure of facial activity detection system.

## 2.1 Detection and tracking of face and ROIs (eyes and mouth)

The general state diagram of the detection and tracking of face and ROIs (eyes and mouth) are represented in Fig. 2.

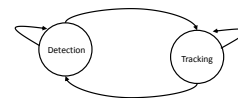


Figure 2. State diagram of the detection and tracking of face and ROIs.

The detection of face and ROIs is performing using the Matlab functionality *vision.CascadeObjectDetector*, which is based on detection by sliding window. The Viola & Jones algorithm [2] is the selected processing, including certain improvements such as rotated Haar characteristics and the use of weak classifiers CART (Classification and Regression Tree) with up to four Haar features. The performing of tracking are made using the KLT algorithm. Fig. 3 shows the result of the detection and tracking subsystem.

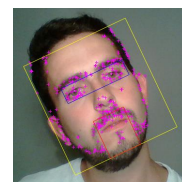


Figure 3. Result of detection and tracking subsystem.

## 2.2 Eyes and mouth state detection

The state detection is performed in each frame, that is, in a static picture.

### 2.2.1 Eyes state detection

Eyes status detection, open or close, has been performed using the model of Bag of Words. This model needs the

creation of a visual dictionary, based on the analysis of labelled pictures with open or close eyes. A visual dictionary has been created, while a SVM classifier has been trained and used to classify (tested with cross-correlation).

### 2.2.2 Mouth state detection

Segmentation method [3] has been selected to perform the mouth status detection. The segmentation method is based on colour, and the YIQ model provides good results in mouth detection [4]. Specifically, channel Q is enough for this task, because the lips and the inside of the mouth (except the teeth) have a high purple content.

## 2.3 Face movements and turns estimation

Determining the attention of a person includes the detection of movements and turns made by the face. This point on, the analysis must be performed comparing images along time, not just individual images.

### 2.3.1 Face movements estimation

The method used compares the detected face regions (eyes and mouth) of the current image with the detected face regions of the closest image in which the face detection stage was performed. To this end, a simple algorithm has been developed based on calculating and comparing the central points of the ROIs that delimit the face.

### 2.3.2 Face turns estimation

The estimation of face turns is based on determining three angles: pitch, roll and yaw. Since the designed system only has one camera, it is necessary to perform 3D orientation from a 2D image. The selected method [5], is based on estimations of the head orientation.

## 2.4 Temporal analysis of eyes and mouth status

Determining activities such as blinking, prolonged eyes closing, yawning, surprise, etc. need to analyze several consecutive images.

### 2.4.1 Temporal analysis of eyes state

Temporal analysis of eyes state has to take into account that the average duration of a human blink is between 50 and 500ms. To this end, the video camera is configured with a frame-rate of at least 60fps. Fig. 4 shows the finite state machine to determine the state of eyes between staying awake, blinking and sleeping. Associated to this state machine is the estimation of the frequency and duration of blinkings. The condition to change to the state sleeping is that the eyes were closed more than 500ms.

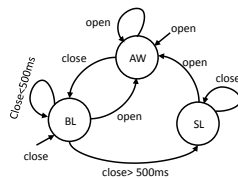


Figure 4. Finite state machine to determine the state of eyes between awaking (AW), blinking (BL) and sleeping (SL).

### 2.4.2 Temporal analysis of mouth status

Since the objectives of the described system are monitoring attention and fatigue, it is crucial to determine if the person has the mouth closed, is talking or yawning. Fig. 5 shows the finite state machine to determine the state of mouth between close, talking and yawning. Associated to

this state machine is the estimation of the frequency and the duration of the yawns. The condition of the yawing is mouth open for over 3 seconds.

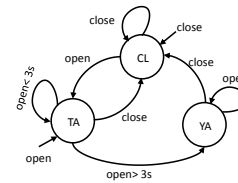


Figure 5. Finite state machine to determine the state of mouth between close (CL), talking (TA) and yawning (YA).

## 3. RESULTS

The designed system has been tested with 30 different people (15 males and 15 females). We have tested the system both together and in isolated stages. \* Detection of face and ROIs. Success rate is near 100%, being the mouth detection the worst case with a success probability of 85%. \* Tracking of face and ROIs. Tracking is lost in 1 out of 16 tests. \* Eyes state detection. Classification accuracy is close to 95%. \* Mouth state detection. Classification accuracy is close to 90%. \* Temporal analysis of eyes and mouth state. They are always detected correctly, but the detection delay is 500ms for eyes and 3s for mouth.

## 4. CONCLUSIONS

In this contribution, we present a facial activity detection system using image processing and machine learning techniques. This system allows detecting if the eyes are open, blinking or closed by being asleep, as well as detecting the mouth closed, talking or open for yawning. This facial activity detection allows monitoring people emotional states, attention, fatigue, reactions to different situations, etc., in a non-intrusive way. Monitoring the facial activity of a person allows us to know if it is necessary to take a break, change the type of music that is being listened to or modify the way of teaching the class.

## Acknowledgments

This work has been funded by Campus de Excelencia Internacional Andalucía Tech-Universidad de Málaga under educational innovation project PIE17-021. This activity was done in collaboration with Servicio de Publicaciones y Divulgación Científica and the department Ingeniería de Comunicaciones, both at Universidad de Málaga.

## 5. REFERENCES

- [1] P. Ekman and W. Friesen, "Measuring facial movement," in *Journal of Environmental Psychology*, 1976.
- [2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition CVPR2001*, 2001.
- [3] C. Solomon and T. Breckon, *Fundamentals of Digital Image Processing: A practical approach with examples in Matlab*. John Wiley & Sons, 2011.
- [4] U. Saeed and J. Dugelay, "Combining edge detection and region segmentation for lip contour extraction," in *International Conference on Articulated Motion and Deformable Objects*, 2010, pp. 11–20.
- [5] G. Mateos, *Procesamiento de caras humanas mediante integrales proyectivas*. Tesis Doctoral. Universidad de Murcia., 2007.

# The Chordinator: An Interactive Music Learning Device

Eamon McCoy, John Greene, Jared Henson, James Pinder, Jonathan Brown, Claire Arthur

School of Music, Georgia Institute of Technology  
jgreen63@gatech.edu, eamon.mccoy@gatech.edu

## ABSTRACT

The Chordinator is an interactive and educational music device consisting of a physical board housing a “chord stacking” grid. There is an 8x4 grid on the board which steps through each of the eight columns from left to right at a specified tempo, playing the chords you have built in each column. To build a chord, you place blocks on the board which represent major or minor thirds above blocks that designate a root (or bass) note represented as a scale degree. In the bottom row, the user specifies a bass (root) note, and any third blocks placed above it will add that interval above the bass note. Any third blocks placed above other third blocks add an additional interval above the prior one, creating a chord. There are three rows above each root allowing either triads or seventh chords to be built. This interface combined with the board design is intended to create a simple representation of chord structure. Using the blocks, the user can physically “build” a chord using the most fundamental skills, in this case “stacking your thirds.” One also learns which chords work the best in a sequence. It provides quick satisfaction and a fun, interactive way to learn about the structure of chords and can even spark creativity as people build interesting progressions or try to recreate progressions they love from their favorite music.

## 1. MOTIVATION

The goal of this project was to create an interactive and educational tool to help people understand foundational harmonic structures in Western music. We wanted to create a product that had a physical dimension (as opposed to a purely digital product or application) with interchangeable pieces that made the design fun and engaging while remaining as simple and intuitive as possible.

The target user for this product is someone without formal musical training but who may be interested in the building blocks of music theory, or curious about the chords and chord progressions in their favorite songs. In our research for this project we were unable to find any interactive, physical products like this.

There are several standalone and online software application systems for understanding the fundamentals of music theory (i.e., notes, intervals, and chords). The drawbacks to these systems is that they tend to be

expensive (e.g., EarMaster [1]), or highly task-oriented and geared towards aiding novice music students (e.g., [www.musictheory.net](http://www.musictheory.net) [2]), or designed for academic use within a structured program (e.g., Musitian [3], SmartMusic [4]). Our product was designed to be more akin to software products such as Captain Chords [5] (a plugin compatible with many DAWs that allows a user to build chord progressions), but with a physical component. We believe this helps with the learning process by combining a tactile response with a visual (color representation of the intervals and orientation of the third blocks) response, and an auditory response.

Another motivation for this project was the potential for our product to contribute to research in music perception and cognition. With this platform of creating chord progressions, we wanted to see if a user’s choice of timbre affects the type of chord progression they make. In other words, does creating chord progressions with a rock guitar sound tend to “pull” the user into creating a more pop-rock chord progression? To test this, our product is designed to allow a user to choose a timbre, and collects data regarding the final settings (all roots, thirds, tempo, and timbre) when a user is finished using the board. We chose five timbres that we thought best represented five common genres of music: strings for classical, electric guitar for rock, autoharp for folk, synthesizer for electronic, and a rhodes organ for jazz, as well as a more neutral or cross-genre sound, the piano.

## 2. DESIGN

Since the crux of our project relies on “stacking” or “building” chords, we needed a compartmental, sectional design. As such, a design similar to that of a fishing “tackle box” was chosen as the basic architecture for the board itself. We limited ourselves to eight columns (one per chord) and four rows (one root “box” and three third “boxes”) for practical reasons; namely, the addition of more columns would have made the board too large or the blocks too small. Since the average phrase of a popular song is typically four measures with one to two chords per measure, eight columns was an appropriate length to capture a single musical phrase.

In the inside, we wanted the spaces for root blocks to be visually different from the third blocks as to make a visual representation of their importance and also to make the distinction between the block types more intuitive. This was achieved in the form of the root note blocks being larger than the third blocks, meaning the spaces for the root note had to be larger than the third notes. As our box was to be an eight-column by four-row grid, we had to create one row larger than the others while keeping the

*Copyright: © 2019 McCoy et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.*



other three the same. Ultimately we created our grid by creating an interlocking, modular “lattice” of MDF wood strips. Finally, we placed this lattice inside the box to achieve the “tackle box” look we were aiming for (see Figure 1).

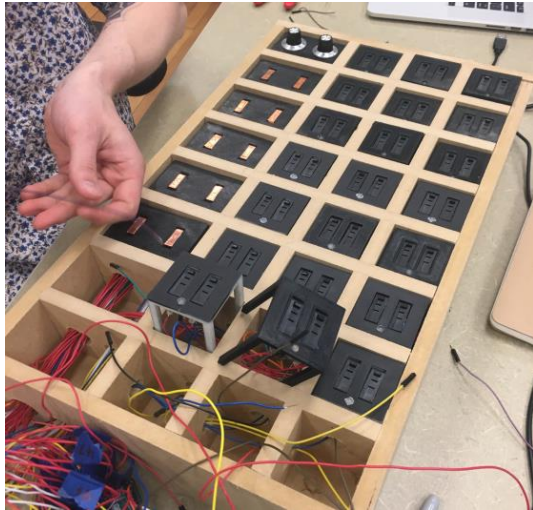


Figure 1: The chordinator board design

For sound generation and switch onset detection we used an Arduino Mega that sends data over serial port to a MaxMSP patch which converts the data to MIDI and plays the chords in real time as they are sequenced. There are two potentiometers, one which controls tempo, and one which controls timbre. As the user creates their chord sequence, they can change instruments and tempo in real time. When they are finished, they press a button and the state of the board is saved.

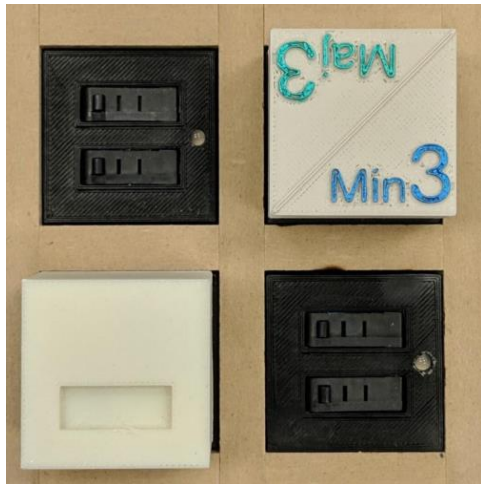


Figure 2: The switch detection method for the thirds.

We decided to use two different systems of communication for our two types of blocks; the third blocks communicate a signal to the Arduino by a simple binary switch depression system, and the root blocks communicate by resistor.

Two switches per block were required to detect three different states: no block, major third, and minor third.

Both switches were aligned on the grid space so that, according to the location of the cavity when placed, it will either trigger one switch or the other (see Figure 2). One orientation corresponding to a minor third interval and the other corresponding to a major third interval.



Figure 3: Resistor detection method for bass notes.

To select the bass note for each chord, the larger (root) blocks are placed on the bottom row of each column. Instead of using switches to detect the different bass blocks, we used different values of resistors and assigned different notes to their corresponding values. Copper tape and a unique resistor was secured on the bottom of each bass block as well as the spaces where the bass blocks are placed (see Figure 3). When the connection is made, the block is identified based on the resistor that was used. From the user's perspective, they are placing a block with a number corresponding to the scale degree. So, in the key of C Major, the number three would correspond to the note E.

### 3. CONCLUSIONS

The Chordinator could be a useful product as a fun, standalone “toy” or game, and has the potential to be modified as a board or game for music education. Its ability to collect data makes it valuable for music perception and cognition research—something we hope to contribute to via this project in the future. We hope to collect user feedback at this stage of the prototype in order to make improvements to any aspect of the design, sounds, or user experience for the purposes of future development.

### 4. REFERENCES

- [1] EarMaster [Software]. (2015). Retrieved from <https://www.earmaster.com/>
- [2] Adams, R. (2018) [Online Software]. Retrieved from <https://www.musictheory.net>
- [3] Rising Software (2019). Musition 5 [Software]. Retrieved from <https://www.risingsoftware.com/musition/>
- [4] MakeMusic (2019). SmartMusic [Software]. Retrieved from <https://www.smartmusic.com/>
- [5] Mixed In Key (2018). Captain Chords Plugin 2.0 [Software]. Retrieved from <https://mixedinkey.com/captain-plugins/captain-chords/>

# Automatic Chord Recognition in Music Education Applications

Sascha Grollmisch, Estefanía Cano

Semantic Music Technologies Group, Fraunhofer IDMT, Ilmenau, Germany

sascha.grollmisch@idmt.fraunhofer.de

## ABSTRACT

In this work, we demonstrate the market-readiness of a recently published state-of-the-art chord recognition method, where automatic chord recognition is extended beyond major and minor chords to the extraction of seventh chords. To do so, the proposed chord recognition method was integrated in the Songs2See Editor, which already includes the automatic extraction of the main melody, bass line, beat grid, key, and chords for any musical recording.

## 1. DESCRIPTION

Gamification of music education has experienced a boost in popularity in recent years due to new possibilities for user interaction enabled by advancements in the field of Music Information Retrieval (MIR) [1]. As an alternative to playing music with printed score sheets, interactive applications with real-time feedback to the users' performance were brought to the market. Some examples of such applications include: Rocksmith<sup>1</sup>, Skoove<sup>2</sup>, Songs2See Game<sup>3</sup>, and Yousician<sup>4</sup>. Besides enabling the analysis of users' performances captured through a microphone, MIR techniques have also enabled the semi-automatic creation of score sheets from music recordings. This has been made publicly available with the release of applications such as Chordify<sup>5</sup>, Songle<sup>6</sup>, Songs2See Editor<sup>7</sup>, and plugins for Sonic Visualiser<sup>8</sup> such as Chordino<sup>9</sup>.

By including the state-of-the-art algorithm for chord recognition proposed by Nadar et al. [2] in the Songs2See Editor, we aim to demonstrate its potential for real-life applications. This highlights the importance of further advancements in the field of MIR to enable non-experts to retrieve complete score sheets of their favorite songs with minimal manual corrections. An example of the proposed integration is shown in Figures 1 and 2.

<sup>1</sup> <https://rocksmith.ubisoft.com>

<sup>2</sup> <https://www.skoove.com>

<sup>3</sup> <https://www.songs2see.com/en/products/game>

<sup>4</sup> <https://yousician.com>

<sup>5</sup> <https://chordify.net>

<sup>6</sup> <https://songle.jp>

<sup>7</sup> <https://www.songs2see.com/en/products/editor>

<sup>8</sup> <https://www.sonicvisualiser.org>

<sup>9</sup> <http://www.isophonics.net/nnls-chroma>

Copyright: © 2019 Sascha Grollmisch et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

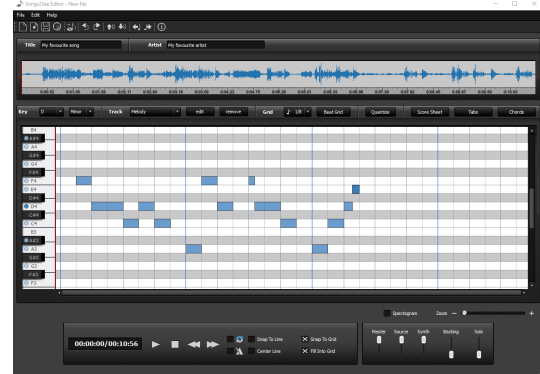


Figure 1. Songs2See Editor interface displaying the extracted melody and beat grid of an imported audio file.



Figure 2. Final score sheet extracted by the Songs2See Editor (including chords) that can be printed or exported to other notation software.

## 2. REFERENCES

- [1] E. Cano, C. Dittmar, J. Abeßer, C. Kehling, and S. Grollmisch, "Music technology and education," in *Springer Handbook on Systematic Musicology*, R. Bader, Ed. Springer, Berlin, Heidelberg, 2018, pp. 855–871.
- [2] C.-R. Nadar, J. Abeßer, and S. Grollmisch, "Towards CNN-based acoustic modeling of seventh chords for automatic chord recognition," in *Proceedings of the International Conference on Sound and Music Computing*, Málaga, Spain, 2019.

# Sonic Sweetener Mug

**Signe Lund Mathiesen**

Aarhus University  
signelma@food.au.dk

**Derek Victor Byrne**

Aarhus University  
derekv.byrne@food.au.dk

**Qian Janice Wang**

Aarhus University  
qianjanice.wang@food.au.dk

## 1. INTRODUCTION

Eating is one of the most sensory of all activities that we take part in. Apart from tasting, it involves both the food and the environment. The multitude of different sensory inputs (from the smell of the food and the colour of the plate, to the lighting in the room and the ambient soundscape) all affect the way we think about and perceive our food [1]. Much like eating, listening is a fundamental part of most lives; and similar to the role of food, music can modulate our feelings, our mood, and our experiences in life.

This demo explores the common link between these two phenomena, specifically the way in which what we taste can be influenced by what we listen to.

## 2. BACKGROUND

Associations between sensory input across all the senses exist, and a substantial body of research on the topic of multisensory perception has shown that individuals experience the phenomenon of “crossmodal correspondences” which can be explained as seemingly arbitrary and unrelated associations between sensory features from different sensory modalities [2]. For example, the combination of a figure spatially positioned upwards and a high pitched sound is often perceived as related and constitutes an example of a crossmodal correspondence [3].

Recent research now demonstrate how music and sound can influence our eating experiences and taste and flavour perceptions. For instance, evidence shows that individuals are able to systematically map sonic attributes (such as pitch, articulation, tempo, harmony, etc) and basic tastes [4]. For instance, high-pitched sounds are often associated with sweet or sour tastes, whereas low-pitch sounds tend to be associated with bitter tastes [5–7]. By making use of such crossmodal correspondences, pieces of music can be chosen or composed to correspond with the taste, mouthfeel, or flavour of a particular food or drink [8]. Furthermore, research within the emergent field of “sonic seasoning” has shown that such soundtracks can actually alter people’s actual taste experience [9].

To further explore the phenomenon of sonic seasoning, we have constructed a system where the way in which

people pick up a mug triggers the onset of different taste-corresponding soundtracks to accompany the drinking experience. This demo turns the act of drinking into a form of embodied interaction with music, while high-lighting how the same beverage can taste very differently depending on the music that happens to be playing at the same time. By building such new interactive systems, we explore the extent to which our distal senses can influence flavor perception, and at the same time use sound to raise people’s awareness of their own eating behavior.

## 3. SOUNDTRACK CREATION

Two soundtracks were created for this demo, one designed to bring out sweetness, and the other to emphasise bitterness. Previous research has demonstrated how high-pitched, consonant, and legato-articulated music is associated with sweet tastes, whereas low-pitched and dissonant music is associated with bitter taste [10–12].

The sweet soundtrack is composed of a fairly high-pitched, slow legato clarinet melody on top of a single C major chord piano arpeggio strum. A string pad is introduced to create a softer timbre.

The bitter soundtrack consists of a low-pitched brass drone with a slightly syncopated cello string figure in the low register. Furthermore, an arpeggiated synth as well as a pizzicato string ensemble adds a percussive element (static pulse) to the track.

Both soundtracks were created in Logic Pro X with the program’s preinstalled software instruments.

## 4. SETUP

The main components and relationships of the *Janus Mug* are schematised in Figure 1. Each of the two handles of the mug is connected to an Arduino microprocessor (Makey Makey LLC) connected to a laptop. When the user picks up one of the handles, they complete an electrical circuit that trigger a specific key press on the laptop. A program built with the Processing software then reads in the key press and plays the appropriate taste soundtrack. This ensures that the user can hear either the sweet or bitter soundtrack as they are drinking from the mug, depending on which handle of the mug they are holding onto. When the user is finished drinking, setting the mug down on the table triggers a different key press, which then stops music playback.

Copyright: © 2019 Signe Lund Mathiesen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

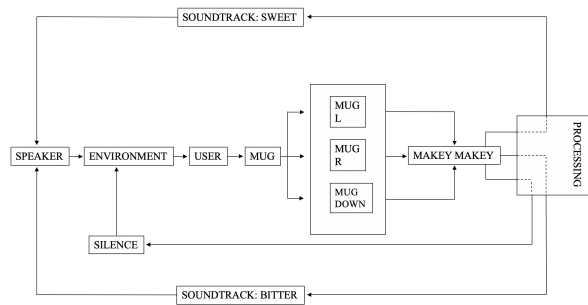


Figure 1. Diagram of the setup

## 5. CONCLUSIONS

Investigating our physical interactions with music while we consume foods or beverages could potentially contribute to the understanding of what drives our eating behaviour. By raising awareness of such behaviours (through the sound/consumer interplay), we can design future sound solutions in a variety of spaces to encourage healthier eating behaviour. Research into “sonic seasoning” can provide food researchers, health professionals as well as restaurateurs with insights into how auditory influences can be implemented in canteens, restaurants, and virtually any context within which eating occurs. Imagine, for instance, how the presence of a sweet soundtrack while drinking coffee could enhance the perceived sweetness of the beverage and thus result in a lower need for added sugar (or cream) to the drink [9, 13].

## Acknowledgments

This work has been supported by the Dean’s start up grant, Faculty of Science and Technology, Aarhus University

## 6. REFERENCES

- [1] C. Spence and B. Piqueras-Fiszman, *The perfect meal: the multisensory science of food and dining*. Wiley Blackwell, Chichester West Sussex UK, 2014.
- [2] C. Spence, “Crossmodal correspondences: A tutorial review,” *Attention, Perception, & Psychophysics*, vol. 73, no. 4, pp. 971–995, May 2011. [Online]. Available: <https://doi.org/10.3758/s13414-010-0073-7>
- [3] E. Ben-Artzi and L. E. Marks, “Visual-auditory interaction in speeded classification: Role of stimulus difference,” *Perception & Psychophysics*, vol. 57, no. 8, pp. 1151–1162, Nov 1995. [Online]. Available: <https://doi.org/10.3758/BF03208371>
- [4] K. Knoeferle and C. Spence, “Crossmodal correspondences between sounds and tastes,” *Psychonomic bulletin & review*, vol. 19, 10 2012.
- [5] A.-S. Crisinel, S. Cosser, S. King, R. Jones, J. Petrie, and C. Spence, “A bittersweet symphony: Systematically modulating the taste of food by changing the sonic properties of the soundtrack playing in the background,” *FOOD QUALITY AND PREFERENCE*, vol. 24, no. 1, pp. 201–204, 2012.
- [6] K. Knoeferle, A. Woods, F. K  ppler, and C. Spence, “That sounds sweet: Using cross-modal correspondences to communicate gustatory attributes,” *Psychology and Marketing*, vol. 32, pp. 107–120, 01 2015.
- [7] Q. J. Wang, S. Wang, and C. Spence, ““Turn Up the Taste”: Assessing the Role of Taste Intensity and Emotion in Mediating Crossmodal Correspondences between Basic Tastes and Pitch,” *Chemical Senses*, vol. 41, no. 4, pp. 345–356, 02 2016. [Online]. Available: <https://doi.org/10.1093/chemse/bjw007>
- [8] Q. J. Wang, A. T. Woods, and C. Spence, ““what’s your taste in music?” a comparison of the effectiveness of various soundscapes in evoking specific tastes,” *i-Perception*, vol. 6, no. 6, p. 2041669515622001, 2015. [Online]. Available: <https://doi.org/10.1177/2041669515622001>
- [9] C. Spence, F. Reinoso-Carvalho, C. Velasco, and Q. J. Wang, “Extrinsic auditory contributions to food perception 5 & consumer behaviour: An interdisciplinary review,” *Multisensory Research*, in press.
- [10] F. Reinoso Carvalho, Q. Wang, R. van Ee, D. Persoone, and C. Spence, ““smooth operator”: Music modulates the perceived creaminess, sweetness, and bitterness of chocolate,” *Appetite*, vol. 108, p. 383–390, 2016.
- [11] F. R. Carvalho, Q. J. Wang, R. V. Ee, and C. Spence, “The influence of soundscapes on the perception and evaluation of beers,” *Food Quality and Preference*, vol. 52, pp. 32 – 41, 2016. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0950329316300532>
- [12] A.-S. Crisinel and C. Spence, “As bitter as a trombone: synesthetic correspondences in nonsynesthetes between tastes/flavors and musical notes,” *Attention, perception & psychophysics*, vol. 72 7, pp. 1994–2002, 2010.
- [13] Q. Wang, L. Mielby, A. Thybo, A. Bertelsen, U. Kidmose, C. Spence, and D. Byrne, “Sweeter together? assessing the combined influence of product-related and contextual factors on perceived sweetness of fruit beverages,” *Journal of Sensory Studies*, p. e12492, 02 2019.



# A FRAMEWORK FOR THE DEVELOPMENT AND EVALUATION OF GRAPHICAL INTERPOLATION FOR SYNTHESIZER PARAMETER MAPPINGS

**Darrell Gibson**

Faculty of Science & Technology,  
Bournemouth University, UK  
dgibson@bournemouth.ac.uk

**Dr Richard Polfreman**

Faculty of Arts and Humanities,  
University of Southampton, UK  
r.polfreman@soton.ac.uk

## ABSTRACT

This paper presents a framework that supports the development and evaluation of graphical interpolated parameter mapping for the purpose of sound design. These systems present the user with a graphical pane, usually two-dimensional, where synthesizer presets can be located. Moving an interpolation point cursor within the pane will then create new sounds by calculating new parameter values, based on the cursor position and the interpolation model used. The exploratory nature of these systems lends itself to sound design applications, which also have a highly exploratory character. However, populating the interpolation space with “known” preset sounds allows the parameter space to be constrained, reducing the design complexity otherwise associated with synthesizer-based sound design. An analysis of previous graphical interpolators is presented and from this a framework is formalized and tested to show its suitability for the evaluation of such systems. The framework has then been used to compare the functionality of a number of systems that have been previously implemented. This has led to a better understanding of the different sonic outputs that each can produce and highlighted areas for further investigation.

## 1. INTRODUCTION

A fundamental problem of synthesizer programming is knowing how to set the parameters to create a certain sonic output. Many synthesizers have a large number of parameters and although having direct access to every parameter (*one-to-one* mapping) gives very fine control of the sounds, it complicates the process of designing new sounds. Alternatively, it is possible to map a smaller number of control parameters to a larger number of synthesizer parameters (*few-to-many* mapping) to reduce the control complexity. One way in which this can be done is to use *interpolation*, where sets of parameter values (“presets”) for known sounds can be assigned in a point-wise manner to the control variables of a suitable controller. Then as the control variables are changed, via the controller, interpolation generates new values for the synthesizer parameters. In this way, it is possible to create sonic outputs that are constrained by the known sounds and the control changes. This provides a mechanism for exploring a defined interpolation space.

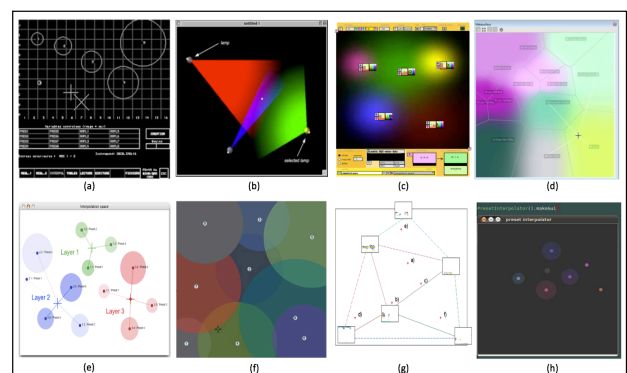
A number of such interpolation systems have been developed, but the systems of particular interest in this body

of work are those that use a graphical interface for the interpolation control. These map presets of synthesis parameters to specific locations in a (normally) two-dimensional pane and the system calculates interpolated parameter values for the *interpolation point* (cursor) position as it moves between the preset locations. This facilitates the discovery of new “custom” parameter values that blend characteristics of two or more parameter presets. The resulting sounds are a function of the interpolation model used, the parameter presets, their locations within the interpolation space, the position of the interpolation point [1] and the synthesis engine itself. It is also possible to define trajectories for the interpolation point that result in new sonic gestures.

Of particular interest here, is the use of such interpolators for sound design, which in this work is taken to be the design of new sounds, often to accompany visual or other media. Sound design is a creative process and as a result there is a desire to remove or minimize any technical barriers between the creative artist and sonic results. A large part of the creative process involves generation and exploration [2], so it is desirable to provide a platform that supports this paradigm effectively.

## 2. PREVIOUS WORK

Over a period of many years a number of graphical interpolation systems have been developed for use with synthesizer/sound processing technology. A summary of these is given in the following sections and an evaluation is undertaken that results in the formulation of a framework. Figure 1 shows the visual representation for each interpolator reviewed.



**Figure 1** Graphical Interpolator Models

Copyright: © 2019 Darrell Gibson. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 2.1 SYTER

Work in this area was first completed at GRM (Group de Recherches Musicale) in the early 1980's on the SYTER system, a hardware workstation that was designed to allow real-time audio processing and synthesis. SYTER had a two-dimensional graphical interface, which offered a user-friendly real-time control window, called INTERPOL [3] to control the relationship between different parameter presets in a real-time sound-processing engine (Figure 1a). The positions of points on the visual interface are mapped to presets of up to 16 parameters. Each preset has a circular representation (a *planet*) in the interpolation space and clicking on a planet recalls the sound of the corresponding preset. However, the system also allows interpolation between presets using a gravitational model, where influence varied with the distance between the planets and their size. In this way, larger planets have a higher gravitational force and so a stronger field-of-influence compared to smaller planets. The work has been expanded over the years for three-dimensional graphical control and a generalized Inverse Weighted Distance (IWD) model [4], where the exponent value can be user controlled.

## 2.2 Interpolator

The SYTER style gravitational model for interpolation was further expanded with a system called Interpolator, which was developed in collaboration between GRM and University of Hertfordshire in the early 2000's [5]. This prototype system was designed as a graphical control interface for the GRM Tools software plug-ins. The system used a light model for the interpolation, where presets were represented as *lamps*, with each having an angle, aperture and extent of the light source (Figure 1b). The light beams gave a visual representation of the corresponding preset's field-of-influence. In addition, if the angle of a lamp's aperture is opened up to 360 degrees then it becomes similar to the planetary system, except the lamp shows the field-of-influence and the planet's area is not lost from the interpolation space. Users could then explore interpolated sounds where the lamp's light beams intersected. Different colours (up to 4) were used to signify different mapping layers for the interpolation. Hence, a colour represented a set of parameters mapped to either single or multiple GRM Tools plug-ins (up to 4) and a lamp is a specific set of values for the parameters. This design allowed layers to be created in the interpolation on the same or different plug-ins.

## 2.3 Gaussian Kernels

In 2003, Momeni defined a system that allowed the spatial layout of objects that relate to musical material – either recorded samples or synthesis parameters. Each of these can be placed at locations within a two-dimensional graphical pane and represent a Gaussian kernel, whose value at any given point in the pane indicate the weight of the associated preset point in the interpolation. This allows weighted interpolation among the preset points based on

the values of the Gaussian kernels at each point in the interpolation space. For each kernel the user could modify the location, amplitude and standard deviation [6]. The interpolation space then shows a two-dimensional visual representation of all the kernels in the space and the kernel's amplitudes are mapped to the brightness scale of a selected colour (Figure 1c). For more accurate visualization the Gaussian kernels can also be viewed as a three-dimensional image. The space created can be explored and interpolation between the presets is calculated based on the cursor position and weight of the kernels. The Gaussian kernels provided not only a mechanism for interpolation, but also for extrapolation beyond the perimeter of the points specified in the space. As this system was implemented in the visual programming environment Max, it is possible to control any sound engine that can be created in it.

## 2.4 Metasurface

In 2005 the Metasurface was developed as a control interface for the AudioMulch Interactive Music Studio, a software application for live performance, audio processing, sound design and music composition [7]. Metasurface can be used to control synthesis and processing parameters and allows any number of parameter presets to be defined and placed in the interpolation space. When the presets are placed in the interpolation space a Voronoi tessellation is constructed where each preset is at the centre of a convex polygon (Figure 1d). Any position contained in each polygon is closer to the centre point of that polygon, than the centre point of any another polygon. Moving the cursor within the tessellated pane performs natural neighbour interpolation. This is calculated by adding a new polygon for the current cursor position and the weight of each neighbour is then calculated as the area "stolen" from the neighbours by the polygon centred at the cursor position. Moving the cursor results in smooth interpolation between the cursors natural neighbour as it moves through the space.

## 2.5 INT.LIB

Work in the mid 2000s saw the SYTER style gravitation model revived, updated and expanded by Oliver Larkin. INT.LIB is a library for Max that allows the control of multiple layered presets using a gravitational model (Figure 1e). Each layer is color-coded and has its own cursor that indicates the location of the interpolation point for that layer [8]. Optionally the interpolation points can be linked so that all layers are controlled simultaneously. Each of the layers has its own instances of a synthesizer or signal processing plug-in and allows interpolation between a number of patches on that sound engine. As INT.LIB is implemented in the Max environment it again means it has open ended possibilities for the synthesis engine.

## 2.6 Nodes

Andrew Benson, a visual artist, created the *nodes* object for Max in 2009 and it proved so popular that it has been included in subsequent Max distribution (Figure 1f). When combined with the *patrstorage* object it can provide a graphical interpolation system. Although the *nodes* system uses a distance-based interpolation function, it uses a different model, where each preset is represented as circular node within the interpolation space. The interpolation is only performed in regions where the nodes intersect. When the cursor is inside a node the distance to the node's centre is used as the weighting for the corresponding preset [9].

## 2.7 Spike-Guided Delaunay Triangulation

In 2009 Drioli et al., developed another graphical interpolation scheme as a sound design interface for physically-based synthesis models. A visual spike representation for the sonic output of each synthesized preset can be positioned in an interpolation pane (Figure 1g). The presets form a scatter of points on the graphical pane and interpolation is performed based on a Delaunay triangulation of the points. The user can select points in the space and the synthesizer parameters are calculated through linear interpolation of the three presets of the containing triangle [10].

## 2.8 Intersecting N-Spheres Interpolation

Developed by Martin Marier in 2012, Intersecting N-Spheres Interpolation is a mapping strategy for interfaces including multiple continuous sensors [11]. This system uses a two-dimensional space where the presets and interpolation point are positioned. The visual representation shows a circle around the interpolation point, with a radius equal to the distance of the nearest preset point. Circles are also drawn around each preset point, with the radii of these circles being equal to the distance to the nearest preset location or the interpolation point, whichever is nearest (Figure 1h). Any preset point circles that intersect the interpolation circle are considered neighbours and influence the interpolation. The value of the interpolation point is calculated as a weighted average of the value equal to the ratio of intersecting circles area. This system is realized in SuperCollider where it can control the audio processing and synthesis parameters running on this platform.

## 3. EVALUATION OF GRAPHICAL INTERPOLATORS

Although the systems examined in Section 2, have all been created to allow interpolated control of parameters they represent different realizations, are implemented with different technologies and have a number of different application areas. Nonetheless, there is a common thread, in that interpolation allows the adjustment of sound parameters between defined presets, via some form of visual model. From the systems examined it is apparent that they can be decomposed into five different, but dependent areas

that should be considered when developing such systems. These are:

1. Control – input controls of the interpolation model
2. Visual Metaphor – the visual interpolation model and how it is represented graphically
3. Interpolation – the interpolation weighting calculations
4. Mappings – the synthesis parameters that are interpolated
5. Synthesis – type/architecture/implementation of the sound engine

Each of these areas will be considered separately, however, for a number of the systems examined in Section 2 there was not a clear partitioning between them. In addition, in this work they will be considered in a sound design context.

### 3.1 Interpolation Control

There have been many different ways of controlling interpolation systems. Here these have been constrained to those offering a graphic interface that corresponds to the visual interpolation model. Many of the older systems had two modes of operation: one for the creating and editing the interpolation space and another for actually performing the interpolated sonic output [2, 5, 7, 12]. This meant that the interpolation space could not be changed in the middle of a sonic exploration, without changing mode. However, if the interpolation calculations and graphics updates can be performed real-time it opens up the possibility of being able to control the interpolated sonic output, either by changing the location of the interpolation point within the space or by modifying the interpolation space itself: moving the preset locations or adding and deleting presets. Moreover, with the layered interpolation system presented in INT.LIB, it is possible to have multiple interpolation points and these can either be moved individually or linked so they can all be moved simultaneously [8].

With the possibility of altering the interpolation space in real-time, it is also worth considering the input mechanism for controlling a graphical representation. Using traditional computer-based spatial control devices (mouse, drawing tablet, joystick, trackball, etc.) only one point can be controlled at a time. This means that only one preset position or the interpolation point can be moved at a time. Whereas with multi-touch screen technology it opens the possibility that sound design can be undertaken by simultaneously controlling multiple points in the interpolation space. Being able to change the interpolation space real-time and multi-touch technology, opens the following potential modes of operation for changing an interpolated sound, creating new possibilities for the control of an interpolated sound design process:

1. Move the interpolation point(s)
2. Change the field of influence for one or more preset
3. Simultaneously move one or more preset locations, while the interpolation point remains static
4. Simultaneously move the interpolation point and one or more preset locations

Discontinuities in the interpolation space are not normally desirable for this application domain, as noted by other authors [7, 12]. However, if the user does want to produce

audible jumps either a new instantaneous position for the interpolation point can be selected (a jump) or if real-time mode is available the position of the presets could be instantaneously changed.

For sound design applications, the addition of performance expressions is often desirable to “bring the sounds to life”, e.g. to match the sound to on-screen actions. It has already been shown that interpolation methods provide an opportunity to apply expressive control to synthesized sounds [12, 13]. However, it does not necessarily follow that expressive control of the interpolation should be performed at the same time as the design of the base sound. It may also be the case that more traditional avenues for applying expressions will still be preferred, for example, through physical actions [14].

### 3.2 Visual Model & Graphical Representation

As can be seen by the range of systems examined, there have been many proposed visual metaphors for graphical interpolators. The visual model provides the user with feedback on the state of the interpolation method, location of the presets and their relative influence. This is delivered in addition to the auditory feedback generated by the synthesizer output. However, for a sound design task it is not clear if the visual representation is needed or actually aids the process.

In the systems examined there are different visual representations for the presets within the interpolation space, often using geometric shapes: circles, triangles, polygons, etc. However, there tends to be some form of visual linkage between these representations and the actual interpolation model. For example, circles have been used to represent presets in a number of different interpolation systems [3, 8, 9, 11], but the way they are interpreted is directly linked to the interpolation paradigm being used in each case. In some, the shapes used in the visualization are linked to which presets are included in the interpolation calculations. For example, where triangulations are generated between the preset locations it provides the implication that the interpolation is being performed between the three presets of an enclosing triangle [10], a rectilinear grid implies interpolation between four local presets and polygons implies interpolation between the closest presets that form a convex hull around the interpolation point [7]. Even a straight line (slider) can be used to imply interpolation between two presets (a 1-D interpolation space). For intersecting interpolation paradigms, where the interpolation is performed when preset objects overlap in the space, the intersection itself implies which presets are included in the interpolation [5, 9]. In other cases, the sounds included in the interpolation are shown by links between the interpolation point and the presets [8].

As well as different geometric representations for presets in the interpolation space, colour is also used in the majority of the systems examined. In most cases the colour is used to differentiate between the presets within the interpolation space. However, in some cases the colours or shadings are visually interpolated to give a visual cue for the interpolated values between the presets [1, 5, 7]. On the multi-level interpolation systems, Interpolator and INT.LIB, colour is used to distinguish between different layers in the interpolation space [5, 8], but the influence of

each preset is provided by linking the visual transparency of the preset’s display colour. In this way, a solid colour shows a preset has a high degree of influence and it becomes more transparent as the influence decreases. This is also the case for the linkage lines that show which presets are included in the interpolation, although the base colour already provides this information.

It is also worth noting that with most of the systems examined the visual representation relates to the interpolation model (*parameter space*) and not the systems sonic output (*sound space*). As seen through the work on *timbre space*, it is possible to use a sound-based representation for the control the synthesis parameters [15]. Although work has continued in this area, for sound design applications, the use of a predefined timbre space may be restrictive. The visual interpolator systems examined do not use automatic positioning of presets within the interpolation space. Instead the user can define the presets that will be used in the interpolation space, the positional relationships between them in the space and in some cases the influence of individual presets. These aspects allow sound designers to constrain the sonic output, while also supporting the exploratory nature of a design process [7, 12].

Finally, with most graphical interpolation systems, individual presets can be recalled by positioning the interpolation point cursor directly on the preset’s position. However, with the SYTER gravitational model, the gravity remains the same while on the planet’s surface so any position on the planet will recall that preset [3]. As a result, the area of each planet effectively reduces the potential size of the interpolation space [5]. Conversely, using the *Max nodes* object, the associated preset can only be recalled by clicking on an area of the node that does not intersect with another node. If a non-intersecting area does not exist then it is not possible to hear the defining sound.

From this analysis, the systems already created have used the following visual cues in the interpolation space:

1. Preset handle (location in the space)
2. Preset field-of-influence
3. Interpolation point(s)
4. Number of presets included in the interpolation
5. Interpolation strength at the interpolation point
6. Navigable space

### 3.3 Interpolation Methods

A variety of methods have been used to calculate the interpolation values between the presets. For example, linear, power, regularized spline with tension, etc. The method chosen will affect the sensitivity and “feel” of the interpolation system, as was demonstrated with LoM [16]. However, it is not clear how the system’s control, visual model, parameter mapping and synthesis engine combine to affect the feel.

As already noted in Section 3.1, it is desirable that an interpolation system should produce a “smooth” sonic output that does not possess discontinuities or overshoots. Therefore, the interpolation function should be smooth to provide even changes and variation to the synthesis parameters and so the sonic output. Although, in some situations jumps maybe required, this should be under user control



and not occur unexpectedly as a result of the interpolation method.

### 3.4 Parameter Mappings

Although the use of interpolation gives the user a mechanism to adjust multiple parameters simultaneously between preset values, the sonic output is defined by which parameters are mapped to the interpolation points. Interpolating all the parameters within a set of presets can create large sonic changes, whereas a mapping that contains a subset of parameters offers more focused control. Moreover, with some forms of synthesis there is not a simple link between the synthesis parameters and the sonic output. As a result, selecting mapped parameters to create a specific sonic result can be difficult. Previous research into the mapping of synthesis parameters has tended to focused on musical outputs and instrument gestural control [12, 13]. This is a different application area and the outcomes have been fairly broad, not considering specific relationships. Multi-layered mappings have been proposed, where intermediate abstract parameters can be used [12], but for interpolation systems being used in musical instrument design. Nonetheless a number of desirable characteristics have been identified for the mappings, such as, differentiability, linearity, range space, exactness, extensibility and editability [13].

With the graphical interpolation systems examined, the mapping between the synthesis engine and the interpolation points is often controlled by the user. This is done by presenting the user with a list of parameters and allowing them to select the desired parameters to map between the visual interface and the synthesis engine. Although this process gives control to the sound designer, completely different sonic outputs will be generated depending on which parameters are selected and those that are not. With the majority of the systems examined one set of mappings is controlled by the graphical interface, however, both Interpolator and INT.LIB considered a multiple mapping approach offering simultaneous control [5, 8]. With INT.LIB, each mapping was sent to a different sound module so different sounds could be layered and controlled separately. Whereas with Interpolator it also allowed multiple mappings to be associated to the same sound module, which allows different aspects of a sound to be controlled independently.

### 3.5 Synthesis

From the range of systems examined, it can be seen that interpolation has been used with many kinds of audio processing and synthesis engines. A number of the earlier interpolation systems are directly integrated into the same platform as the synthesis engine. This means that although the sound can be changed within the remit of the given synthesis engine, it is not possible to use the same interpolation platform with a different synthesis engine. The later exceptions to this have been developed through programming environments [3, 8, 9]. The flexibility of using the programming environment means that it is possible to build new synthesis engines to be used with the interpolation system. Moreover, as the Max environment also supports use of common audio plug-in formats, it is possible to use many commercially available software synthesizers with interpolators built in Max [4, 8].

An interpolator user interface can mask the details of the synthesis and the associated parameter manipulation from the user,

allowing the sound designer to concentrate on the design process, without having to worry about the underlying details of the synthesis engine.

## 4. GRAPHICAL INTERPOLATION FRAMEWORK

Although a number of graphical interpolation systems have been created and documented, they were developed over a thirty-five-year period, using different implementation platforms, different synthesis architectures and were designed for different application purposes. Consequently, many of the realizations used technologies that are now obsolete and no longer available making it impossible to do back-to-back evaluation between the original systems. In order to be able to evaluate the suitability of these graphical interpolation systems for the purpose of sound design, they require re-implementation on contemporary hardware and software platforms. This will allow direct comparisons to be undertaken between the different interpolation systems.

It is important to also consider the characteristics that a sound design graphical interpolator should ideally possess. The following summarizes the most important factors from the evaluation section:

1. Synthesis independent interpolation – the same interface can be used with different synthesis engines
2. Clear relationship between interpolation control and the sonic output – sound space defined by the populated parameter preset
3. Constrain the navigation and exploration of the parameter space – user selecting and positioning presets in the interpolation space
4. Control a number of parameters simultaneously – reduce the control complexity of many parameters
5. Changeable parameter mappings – provide user with control over the parameter mappings
6. Exploration of the sound space with both course and fine levels of detail – change resolution and precision
7. Smooth interpolation – no discontinuities unless user selects
8. Real-time interpolation (not different edit/interpolate modes) – allow either preset points or cursor to be moved to change sounds
9. Support the design of base sounds and the application of performance expressions
10. Usability, repeatability, predictability and playability – user can design a sound based on the supplied preset sounds.

In order to be able to evaluate these aspects of different graphical interpolators a hierarchical framework is proposed that compartmentalizes each of the system elements. This works from the control input at the top-level to the sonic output at the bottom, as shown in Figure 2. Although the final output, sound, is at the bottom level it is worth noting that the visual representation also gives the user visual feedback on the current configuration of the interpolation system and therefore, the sound. Equally the user maybe given inputs that allow the mappings to be modified. However, what the framework shows is the interdependencies of the different elements of an interpolation system and the relationships between them. For example, the

sonic output from the synthesizer is dependent on the control inputs, the visual model, the interpolation function, the parameter mapping and the synthesis engine used. In addition, it is envisioned that in the future different realisations may be created that will still be encapsulated by the framework.

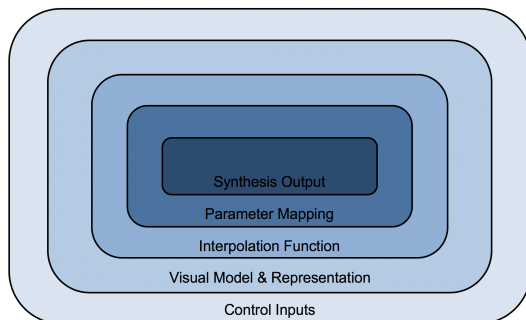


Figure 2 Graphical Interpolation Framework

#### 4.1 Framework Structure

Having formalized the framework, the next stage was to consider an implementation. Using the framework defined in the previous section, it is possible to structure the different levels (control, visual model, interpolation, mappings and synthesis) into separate modules and test them separately. In this way, it becomes possible to directly compare these aspects of each system and evaluate their impact on the usability. This can be done through comparative user tests where only one element is changed at a time. The results can then be measured, compared and evaluated to determine the suitability of each for sound design applications. To facilitate this the framework has been implemented in the Max environment using the architecture shown in Figure 3.

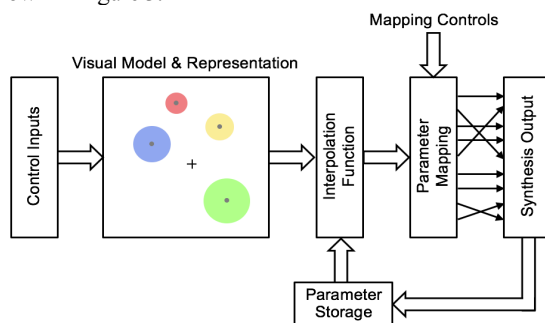


Figure 3 Framework Architecture in Max

#### 4.2 Framework Implementation

As an initial investigation, a graphical interpolation system was built in Max using the *nodes* object detailed in Section 2.6. In this way, the nodes graphical interpolation system acted as proof-of-concept for the framework defined. When this interpolator system was implemented, care was taken to develop each of the five elements of the interpolation framework into separate entities. This was done through a modular design approach where each part is created as a separate module so that each can be modified independently of the others.

The first implemented was the **interpolation function** module, which is storage that holds the parameter values and performs the interpolation. The parameter values for

each synthesis preset are stored as a new data set and it then interpolates between the parameter data sets, generating interpolated values for all the individual parameters. The interpolation is performed based on the modules input which is the relative weightings for each preset. By default, the calculation performed is linear interpolation, but it is possible to change the mode so that any interpolation function can be realized.

As the *nodes* object has been specifically designed as a graphical interpolator, the object has been created with specific functionality for the **visual model** and the **control inputs**. The **control inputs** realized in the *nodes* object are standard computer-based spatial controls. However, it is also possible to send the object positional input data from other sources. This provides the possibility of using other input devices to control the interpolation space. The interpolation point on the *nodes* object can be moved within the space and an output weighting for each node is generated. The **visual model** generated node weights are normalized (0.0 – 1.0) and are proportional to the interpolation point's distance from the circumference of a containing node to its centre. Therefore, when the nodes in the interpolation space overlap and the interpolation cursor is placed in an overlapped region, a weighting is generated for each node. (In Figure 4 - 1 = 24%, 2 = 0% & 3 = 76%).

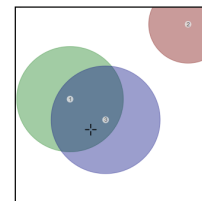


Figure 4 Nodes Outputs Normalized Distances

These weightings are used as the input to the **interpolation function**. As the visual interpolation model is encapsulated by a single object (*nodes*) it is possible to replace it with different implementations.

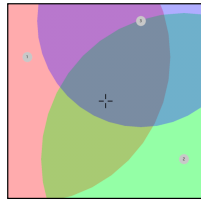
The **synthesis engine** has been constructed to be separate from the interpolation platform by using software plug-ins, allowing different (commercially available) synthesis engines to be loaded and tested. However, the framework would also allow bespoke synthesis patches to be used. When a new synthesizer is loaded, it is interrogated to determine all the parameter values for the number of presets loaded. Each preset is associated to a node in the interpolation space and all of the preset's parameter values are sent the **interpolation function** storage.

By default, all of the parameters for the presets are associated to the corresponding node and so every aspect of the sounds synthesis is controllable. However, the **parameter mappings** between the **interpolation function** and **synthesis engine** can be changed by user selection.

##### 4.2.1 Framework Testing

The prototype nodes-based interpolator was initially tested to ascertain if each module built in the framework could be changed independently of the others and to establish the impact on sound design tasks. Through exploratory testing, where the nodes-based interpolator and its parameter space were left the same (shown in Figure 5), it became apparent that changes to each module in the framework

leads to the system generating different sonic outputs and results in a different user experience with each realisation.



**Figure 5** Nodes Prototype Test Layout

From testing with different **synthesis engines**, it was found that changes to the engine (preset changes, synthesis realisation or changes of synthesis type), were the main determinants of the sonic output. Moreover, with some forms of synthesis, changes to a single parameter can produce large sonic variations, but for others, more subtle alterations resulted. Changes to the **control inputs** allowed different mechanisms for interacting with the sonic manipulation, and potentially changing the usability of the interpolator. Modifications to the **parameter mappings** permitted the refinement of the sonic changes that it is possible to generate with the interpolator. Mapping lots of the synthesis parameters to the nodes resulted in big sonic changes, whereas mapping a few parameters permitted more subtle variations to be generated. Changing the **interpolation function** resulted the subtlest differences. The chosen function affects how the sound transitions as the interpolation point is moved between preset locations.

### 4.3 Graphical Interpolator Implementation

The prototype nodes-based interpolator was used as the basis for the subsequent development of different graphical interpolation systems. For each **visual model** and its control, the *nodes* object was replaced with an interactive user-interface built using OpenGL for the interpolation model's visual representation and JavaScript to create the control mechanism and calculate the preset weightings. Each model was constructed and integrated with the other elements of the framework for testing. To-date six interpolators have been built, integrated with the framework and functionally tested. These are:

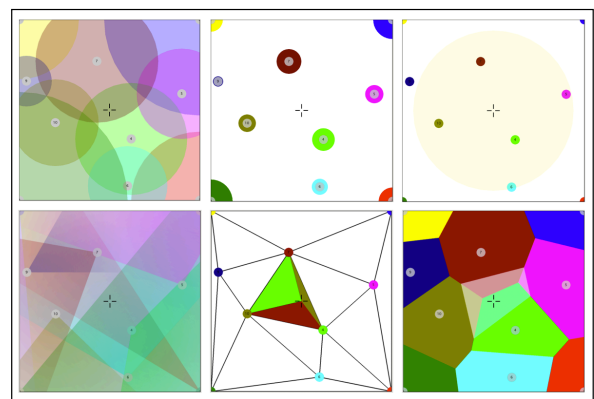
1. Nodes (Overlapping Circles)
2. Gravitational (Planets & Space)
3. Radius-based IWD (Scatter Points & Interpolation Point Circle)
4. Light (Lamps)
5. Delaunay (Triangulation)
6. Voronoi Tessellation (Polygons)

The nodes interpolator was reimplemented so that it could act as a benchmark for the other interpolators, but also so the visual representation can be changed to assess the influence of different visualisations using the same interpolation model. The other interpolators were chosen to represent the key traits of the interpolation systems that have been previously created.

#### 4.3.1 Graphical Interpolator Testing

Following functional testing the different interpolators were back-to-back tested by placing the same ten presets

at identical locations in each. The nodes interpolator was populated first and although the size of each node was randomly selected, they were chosen to ensure the whole space was covered. For the gravitational interpolator, while the same locations were used, this model requires space between the planets, where the interpolation is performed. However, so that each preset has the same relative influence as they do in the node interpolator, the sizes were scaled by one tenth of those in the nodes interpolator. For the radius-based IWD the interpolation point's radius was chosen to cover approximately 50% of the interpolation space so for all interpolation positions, multiple presets are enclosed by the radius. For the light interpolator although the same locations were used, as each lamp has an angle and aperture, it results in each lamp having a specific directionality. To try and give coverage over the whole interpolation space the extent of each lamp was scaled to four times the nodes size. Despite this the lamps directionality also needed to be selectively chosen to ensure the whole interpolation space was covered, whilst still giving a good spread of intersecting light beams. For the two remaining interpolators the presets do not have different influences or directionalities so the locations were kept the same as the nodes layout. The test layouts for the six interpolators are shown in Figure 6.



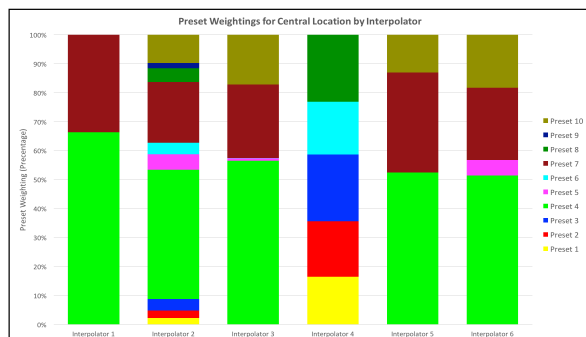
**Figure 6** Test Layout for Graphical Interpolators

These layouts were used to perform back-to-back tests where output from the different graphical interpolators were compared. For the tests the **control inputs**, **interpolation function**, **parameter mappings** and **synthesis output**, all remaining the same, as detailed:

1. Control Inputs – Fixed 2-D movement of interpolation point only
2. Interpolation Function – Linear interpolation
3. Parameter Mappings – All synthesis parameters mapped to the corresponding preset location
4. Synthesis Output – Native Instruments *Massive* with ten presets loaded

The tests compared the sonic output from the different interpolation models for the same interpolation positions. This was first done by instantaneously moving the interpolation cursor to ten different locations and comparing the sound generated with each system. From this test, it was evident that each visual interpolator generated significantly different sonic results, despite being populated with

the same preset sounds. To try and get a better understanding of each system's sonic nature, another comparative test was created, where the interpolation point was moved through a fixed trajectory path around the defined interpolation spaces. The path began at the centre of the space, moved diagonally towards the left-top corner until the mid-point and then moved around parallel to the outside edge of the space. It was found that each interpolator gives a very different range of sonic outputs across all interpolation positions. The fact they were different was not necessarily surprising, but the diversity of the sonic differences was not anticipated. Moreover, each interpolator results a completely different sonic palette that it can generate, meaning it is very difficult to create the same sound with each interpolator. This is because each interpolation model results in different preset weightings for the interpolation function. As an example, Figure 7 shows the preset weightings for just the centre position of each interpolation spaces, as shown in Figure 6.



**Figure 7** Comparison of Interpolator Preset Weightings's

In all cases, the relative positioning (layout) of the presets determines the interpolated outputs. Different layouts of the same presets results in different outputs being obtained. It was also noted that for interpolators 1, 2 & 4 the extent (size) of each preset, further changes the interpolations space. Also, the directionality of the lamps in interpolator 4 gives an added element for further modifying the interpolation space. For interpolators 3, 5 & 6 the influence of each preset is potentially the same, but the layout determines the relative strengths. However, for interpolator 3 this is constrained by the interpolation point's radius that determines which presets will be included. If the radius size is changed, corresponding presets will be added or removed from the interpolated output. Whereas interpolator 5 uses only the three closest presets and interpolator 6 uses the natural neighbours.

## 5. CONCLUSIONS

The framework presented has been shown to provide a suitable platform for the testing and evaluation of different graphical interpolation systems. The modularity of the framework components means that each can be modified independently of the others, offering a suitable mechanism for performing formal comparative user testing. In this way, the use of the framework has led to a more detailed understanding of different interpolation models and the identification of where and how sonic differences are obtained. From the testing that has been undertaken so far

three areas have been identified for immediate further investigation. The first of these will be to undertake formal user testing to assess the level of feedback provided to the users by the visual representations of the interpolation model. The second will be to undertake formal user testing to evaluate the suitability of the presented interpolators for sound design applications. Finally, as different synthesis engines reactions to interpolation can be drastically different this will also be examined further.

## 6. REFERENCES

- [1] J.J. van Wijk and C.W. van Overveld, "Preset based interaction with high dimensional parameter spaces," In *Data Visualization*, 2003 (pp. 391-406).
- [2] T. I. Lubart, "Models of the Creative Process: Past, Present and Future," *Creativity Research Journal*, 2001 Oct 1;13(3-4):295-308.
- [3] T. Todoroff, "Control of digital audio effects," *DAFX: Digital Audio Effects*, 2002:465-97.
- [4] T. Todoroff and L. Reboursière, "1-d, 2-d and 3-d interpolation tools for max/msp/jitter," In *Proc. ICMC'09*, 2009.
- [5] M. Spain and R. Polfreman, "Interpolator: a two-dimensional graphical interpolation system for the simultaneous control of digital signal processing parameters," *Organised Sound*, 2001 Aug 1;6(02):147-51.
- [6] A. Momeni and D. Wessel, "Characterizing and controlling musical material intuitively with geometric models," In *Proc. of Conf. on NIME*, 2003.
- [7] R. Bencina, "The metasurface: applying natural neighbour interpolation to two-to-many mapping," In *Proc. of Conf. on NIME*, 2005, (pp. 101-104).
- [8] O. Larkin, "INT. LIB-A Graphical Preset Interpolator For Max MSP," *ICMC'07: Proc of ICMC*, 2007.
- [9] "nodes," *Max Reference*, Cycling 74, 2018.
- [10] C. Drioli, P. Polotti, D. Rocchesso, S. Delle Monache, K. Adiloglu, R. Annies and K. Obermayer, "Auditory representations as landmarks in the sound design space," In *Proc. of SMC Conf.*, 2009.
- [11] M. Marier, "Designing Mappings for Musical Interfaces Using Preset Interpolation," In *Conf. on NIME*, 2012.
- [12] C. Goudeseune, "Interpolated Mappings for Musical Instruments," *Organised Sound*, 7(2):85-96, 2002.
- [13] A. Hunt, M. Wanderley, and M. Paradis, "The Importance Of Parameter Mapping In Electronic Instrument Design," *Journal of New Music Research*, Volume 32, Issue 4, page 429-440, 2003.
- [14] K. Gohlke, D. Black and J. Loviscach, "Leveraging behavioral models of sounding objects for gesture-controlled sound design," In *Proc. of 5th International Conf. on Tangible, embedded, and embodied interaction*, 2011 Jan 22 (pp. 245-248). ACM.
- [15] D. L. Wessel, "Timbre space as a musical control structure," *Computer Music Journal*, 1979 Jun 1:45-52.
- [16] D. Van Nort and M. Wanderley, "The LoM Mapping Toolbox for Max/MSP/Jitter," In *Proc of ICMC*, 2006 (pp. 397-400).



# COMPOSING WITH SOUNDS: DESIGNING AN OBJECT-ORIENTED DAW FOR THE TEACHING OF SOUND-BASED COMPOSITION

**Stephen Pearse**

University of Portsmouth  
stephen.pearse@port.ac.uk

**Leigh Landy**

De Montfort University  
landy@dmu.ac.uk

**Duncan Chapman**

Independent Composer  
dchapmanhoot@gmail.com

**David Holland**

De Montfort University  
dholland@dmu.ac.uk

**Mihai Eniu**

University of Portsmouth  
mihai.eniu@port.ac.uk

## ABSTRACT

This paper presents and discusses the Compose With Sounds (CwS) Digital Audio Workstation (DAW) and its approach to sequencing musical materials. The system is designed to facilitate the composition within the realm of Sound-based music [1] wherein sound objects (real or synthesised) are main musical unit of construction over traditional musical notes. Unlike traditional DAW's or graphical audio programming environments (such as Pure Data, Max MSP etc.) that are based around interactions with sonic materials within tracks or audio graphs, the implementation presented here is based solely around sound objects. To achieve this a bespoke cross-platform audio engine known FSOM (Free Sound Object Mixer [2]) was created in C++. To enhance the learning experience, imagery, dynamic 3D animations and models are used to allow for efficient exploration and learning. All tools within the system are controlled by a flexible permissions system that allows users or workshop leaders to create sessions with specific features based on their requirements. The system is part of a suite of pedagogical tools currently in development for the creation of experimental electronic music.

## 1. INTRODUCTION

The Compose with Sounds (CwS) software package (see Fig. 1) was born out of two distinct ideas and projects. Landy's highly influential *Making Music with Sounds* [3] made strides in attempting to fill the enormous gap in literature regarding sonic creativity for novices and school teachers. For several years, Landy had been working on the ElectroAcoustic Resource Site (EARS [4]). In 2006 the EARS website EARS was supported and adopted by UNESCO becoming a node of their DigiArts programme. After a period of collaboration with the body, they requested 'an EARS for kids'. This subsequently became EARS 2 [5,6]. As children prefer to be working actively whilst learning as opposed to being solely fed information, the EARS 2 pedagogical site needed to be highly interactive.

Furthermore, it demanded a creative dimension that could not be integrated into the site. This provided the opportunity to develop a bespoke software package, not only for EARS 2 but for any inexperienced user in the realm of sound-based composition also known as electroacoustic music.

For many years members of the project team have been working across a wide variety of sound projects in educational and community settings. The target age group for much of this work has been UK Key Stage 3/international middle school students aged 11-14. Within these contexts, it has become rather clear that there were no real software packages that present a simple way of working with recorded audio that does not require specialist knowledge or experience on hand. Projects such as Sonic Postcards [7], typically resorted to using Audacity as an audio editor and composition environment. The main drawback of this was that it was very hard for those without experience to fully comprehend what they were working on. Using Audacity also assumed that users were able to navigate their way round the computer to find the sounds they were going to use. In this setting, participants would often get "lost" and lose interest. Software like Audacity presents a vast number of "choices" regardless of the level of experience of the user. Similarly, when participants have been presented with other "entry level" tools (such as Garage Band) large numbers of them would spend the majority of their time "auditioning" sounds and presets, leaving little time to actually compose with them.

The CwS software was subsequently designed to be as intuitive as possible while avoiding the interface or auditioning shortcomings of existing tools. The core target age group of the work remains students aged 11-14 but it is flexible and approachable enough to be used by students either side of this age bracket. The software aims to support students to a level wherein they can easily move onto other platforms such as Logic, ProTools or even Max MSP and the like, while being easily linked to challenges within EARS 2 projects. Like the aforementioned platform, the software is intended to be multilingual for adoption in cultures around the globe. CwS was first supported by an EU Culture grant, 'Composing with Sounds' with further development being supported by a Creative Europe grant for the 'Interfaces' project [8,9].

Copyright: © 2019 Stephen Pearse et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

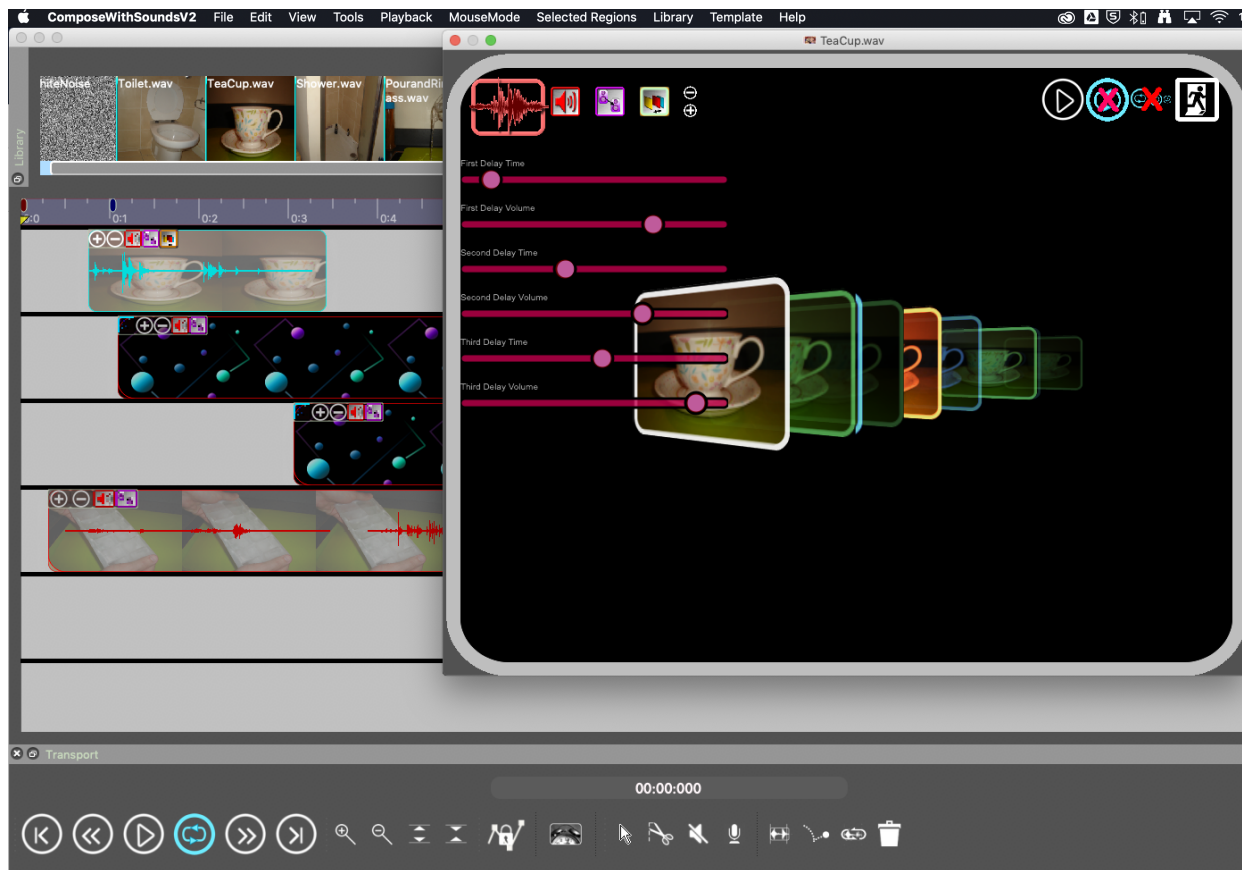


Figure 1. Compose with Sounds Version 2.36

To increase access and potential engagement with the tool and the EARS 2 platform on the whole, the system is scheduled for free release in late spring of 2019 for MacOS 10.9+ and Windows 10.

## 2. TECHNICAL OUTLINE

To enable the development of CwS, a bespoke audio engine known as the Free Sound Object Mixer (FSOM [2]) was engineered. This open-source audio engine was designed to be flexible and suit different iterations of the Compose With Sound project but also be easy to repurpose for other real-time audio applications. With a project of this scale great care and attention needed to be taken to ensure that the functionality in FSOM was not dependent on the CwS in any shape or form. While keeping the audio and graphic threads separate is widely encouraged in the audio development community, here it is fundamental to the architecture of the project.

Throughout the project, CwS has gone through two significant iterations which entailed a complete rewrite of the graphical user interface. Versions 1 through to 1.35, all utilised the cross-platform windowing library wxWidgets [10] and was made available explicitly for Windows 7 and MacOS 10.7 in 2015. To ensure future support for higher resolution displays and stability on newer versions of MacOS, the entire graphical user interface was re-written to

utilise the popular windowing library, Qt [11]. For clarity and integration with other iterations of the software and other projects, all materials that are saved by the system (be it the session itself, library information or template data) are all saved in an easy to read and interpret XML schema.

### 2.1 Development Cycles

To tailor CwS to the target audience in the best way possible and ensure that it be delivered efficiently, an adapted form of the SCRUM development methodology [12] has been used. Software development iterations are based on development sprints lasting between two to four weeks. This flexibility is a necessity as testing and feedback across multiple levels is continuously required to ensure stable development of the software. Since commencing the development of version 2.0, the codebase has been maintained by a single software developer (the lead author). Due to the size of the codebase and the aims of the project, several iterations of testing and feedback take place to identify bugs and feature requests in each private release prior to wider distributions. The software and each new feature is subsequently tested by three distinct groups, each with finite roles (see Fig. 2). These groups are the developer and students at their institution, the wider project team across the interfaces project and workshop participants. The developer is responsible for testing new functionality; the wider team, bugs and general usability issues

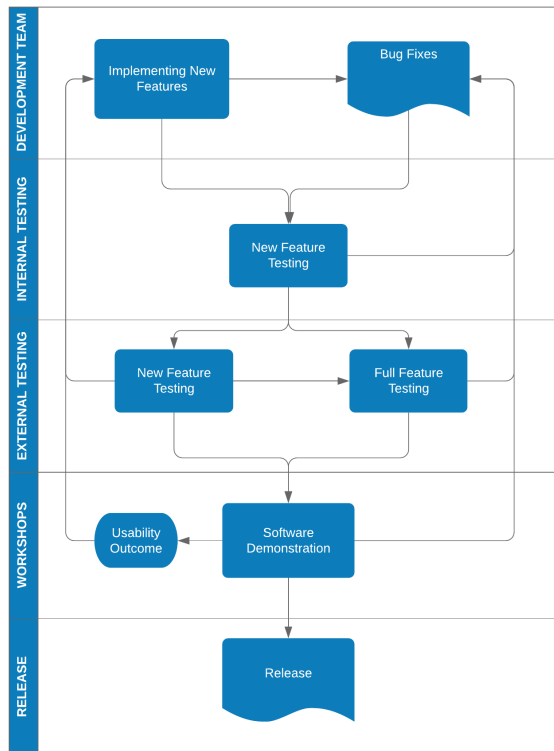


Figure 2. CwS Testing Cycles

and workshop attendees for broader usability issues alongside requesting new features. This approach ensures that if issues or queries concerning the usability emerge, workshop leaders and testers could be informed and guided to test and query functionality.

Many features within the software were subsequently informed through this approach. Perhaps the most notable of these are the graphical tracks alongside the mute and solo tools. As the system does not depend on a traditional track-based architecture (see section 5), sounds can be placed over one another in the sequencer environment with both being heard. Conceptually several users struggled with this methodology and leading to confusion with sounds of different lengths being laid over the top of one another. The software dynamically sorts and alters the rendering z depth of each sound based on their duration with the shorter sound materials being drawn on top of those with longer durations. While this is effective, user feedback indicated that limited graphical “tracks” would be needed to ease the organisation of sessions. After a series of workshops in the summer of 2018, many users requested that these “tracks” offer solo and mute functionality. While this was possible to achieve within the codebase, it was not in keeping with the sound-based nature of the project. As such, mouse tools were created instead for muting and soloing multiple sounds at once within the system.

### 3. THE SOUND CARD

With sounds being affiliated with imagery, the metaphor of a card or soundcard is used within the software. While

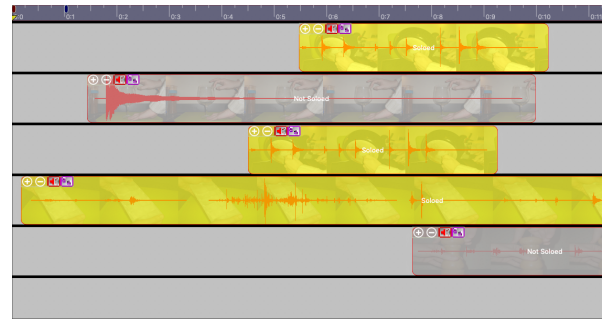


Figure 3. Solo and mute functionality

the pairing of image to sound in this form is encouraged, it is by no means compulsory. At any given moment, a user can alter the view of cards within the system to show the waveform of the sound, the assigned image or both.

A composer’s primary means of working within the system is the sequencing and transformation of these cards. The software subsequently contains a suite of standard editing tools to be utilised. This includes the aforementioned solo and mute tools (see section 2), deletion, splicing, truncation, file bouncing and time-stretching<sup>1</sup>. For ease of use, each of these can be undone and applied to multiple cards at any given time. As well as supporting copy and pasting of cards as a user would expect, the system affords different approaches to duplicating and looping cards. Standard duplication is supported wherein a new card is created immediately after the original has finished. When multiple cards are selected at once, the newly duplicated cards commence at the end of the final original card. These new cards retain the same timing differences as the original selections. Cards can also be duplicated to behave like traditional looping functions found in commercial DAW’s wherein cards are repeated upon their own completion. This flexibility in duplication subsequently enables users to easily create structural relationships based on relative timing or more rigid loops if desired.

#### 3.1 Synthetic Cards

Alongside cards made from audio files on the user’s computer, the system also contains synthetic/generative cards that can be utilised. At the time of writing, these cards consist of a noise card, an additive synthesis card and a granular synthesis card. The number of controls available for the latter two types was purposefully limited to reduce the threat of creative paralysis when confronted with an extensive collection of controls. The former is limited to four oscillators with pitch, amplitude and shape controls (sine, cosine, noise, square, saw and triangle). The granular synthesis region is based on traditional file-based granular synthesis [14] over live input based tools akin with Mutable Instrument’s Eurorack Module, Clouds [15]. This card subsequently provides controls for the grain size, pitch, position in the source file, the rate of spawning, amplitude and playback speed (used to scroll through the sound at varying

<sup>1</sup> The time-stretch tool is based around a custom phase vocoder inspired by the algorithm used in PaulStretch [13]

rates). While random modifiers could have been provided for each of these controls, they have been purposefully removed at this point.

### 3.2 Card Libraries

As the system is based around the usage of sonic cards, the traditional approach of using an audio pool has been replaced by the use of card libraries. This library system is designed to utilise a clear and concise XML schema alongside a simple file structure rather than serialising the data into binary. While previous research [16] has shown the inefficiency of XML under certain circumstances, it has been utilised here to ensure easier integration with affiliated projects and tools. The system allows users to easily import and export libraries either as standalone entities or as part of saved compositions. A selection of card libraries is provided with the software. A further collection of libraries can be found on the original CwS website [17] and the EARS 2 platform [5, 6] which contains teaching materials for the system. Most significantly, it affords the potential for users to interactively explore the libraries on the accompanying website.

At present, the software does not allow users to record sounds within it. This was an active decision made by the team to ensure that the software is used primarily for sound sequencing and transformation. A proof of concept library creation application for desktop and mobile applications has been created using JUCE [18]. This tool allows users to create cards through recording or importing audio and photos on their device of choice. Libraries can then be exported from this tool.

### 3.3 Templates and Levels

The system contains a flexible permissions framework that enables or disables access to features within. Akin with Boden and later Magnusson's work on constraints [19, 20], limiting or guiding users through a limited set of tools (initially) forces them to think creatively about what each tool affords. This subsequently encourages them to internally map out creative possibilities that they wish to explore. The approach taken in CwS supports this further as it enables educators to create their own scaffolding structures [21] for audio tools and effects. These structures are known as templates within the software and can be freely created by the user. This augments Bigg's constructivist theory [22], where users can actively seek and creatively apply their own pathway through the tools available. Templates can be applied to any compositional session or library that the system is aware of. When the student in question is ready, educators can subsequently move to the next relevant template. The system is provided with three linear templates that can be used. An outline of what is provided at each level can be seen in Table 1. To aid the delivery of workshops and other guided learning sessions, templates are saved into libraries when they are exported if required. This enables workshop leaders to design sessions that prescribe the sound materials and tools that can be easily deployed.

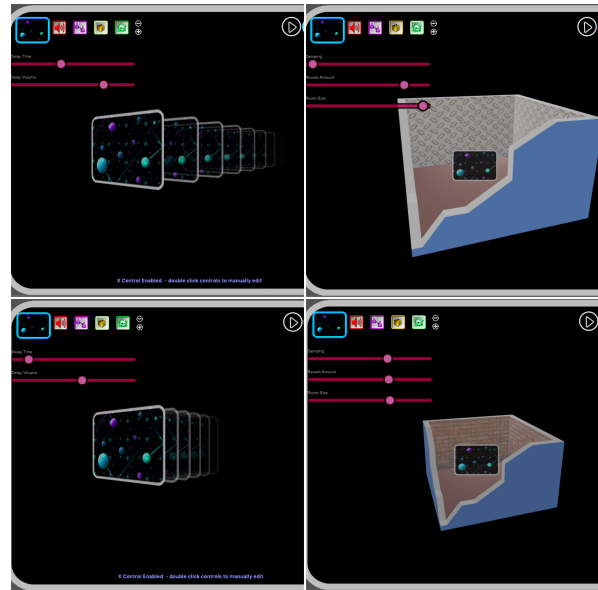


Figure 4. Delay and Reverb representations

## 4. DIGITAL AUDIO EFFECTS AND TRANSFORMATIONS

The system contains a collection of standard and extended audio effect processors (see Table 1) that composers of sound-based music would expect to find in a DAW of their choice or would have access to through commercial/free audio plugins. The system does not allow external plugins to be loaded into it for several reasons. In a classroom context, deploying, loading and working with plugins can slow down sessions due to the risk of technical or licensing issues occurring. This is especially the case in today's climate where some plugins require online activation or verification at runtime. When learning a new piece of software alongside a new approach to music and sound, there is the potential for any student to be overwhelmed with the quantity of features available. If this experience requires users to learn the control and interface vocabularies of different plugin manufacturers on top of this, the chance of creative paralysis may increase. Based on feedback from members of the project team, it was decided that a suite of effects should be designed with a consistent interface to ease students development.

Unlike traditional track-based DAW's, each of the effects contained within the software exists entirely on a sound card<sup>2</sup>. To aid the learning experience when using these effects, each effect processor is accompanied by an interactive 3D animation that updates based on the effects settings (see Fig. 4). The animations provided utilise the card metaphor and presents visible transformations that symbolically, and in some cases, literally reflect the effect process. Effects such as reverb and the collection of delay based effects evidence this divide. The reverb effects present the card within a virtual room where the size, brightness and wall texture alter based on parameters such as room

<sup>2</sup> Automation for parametric changes subsequently exists within a sound card itself.



Level 1	Level 2	Level 3
Audio	All Level 1 features	All Level 2 features
Band Pass Filter	Band Reject Filter	Additive synthesis
Delay	Distort	Asymmetric Delay
Fade	Envelope	Automation
Gain	Harmoniser	Chorus
High Pass Filter	High Amplitude Modulation	Flanger
Loop	Low Amplitude Modulation	Granular Synthesis
Pan	Low Frequency Modulation	Ring Modulation
Reverse	Timestretch	
Simple Reverb	White Noise	
Transpose		
Truncate		

Table 1. Learning Levels by Template in CwS

size, wet/dry and damping. Delay-based effects present repeating cards drifting into the distance based upon the algorithm being used. Across all of these, the delay time is represented as the distance between the cards and the feedback alters changes in brightness across this spread. The asymmetric (multitap) delay presents each tap with a card whose colour palette has been altered while the chorus and flanger both update the separation of the cards based on the modulation source in real-time.

## 5. SEQUENCING

The core audio processing architecture within the system deals with discrete regions. These regions contain all of the audio material for a given card in the software. This approach is unlike traditional DAW designs that will process audio as an array/vector of tracks, each with content within, or as an audiograph with audio nodes that are connected. As there is the potential for hundreds of discrete sounds being active at any given point, careful optimisations are needed to ensure consistent performance on computers with widely different specifications. While every region could be stored in some form of an array with each deciding when they should process their audio (based on the current play-head position and whether it is bypassed), this is dependent on a large number of conditional tests and instructions per sample. While this might not be a problem on contemporary hardware, it may introduce a substantial performance bottleneck on older machines that may be found in schools. To overcome this, the audio engine creates a cache of all of the region based events within the current session. Each event has a type, the region it is associated with along with the sample time that it occurs. In the real-time audio thread, a list of active regions is utilised. Only regions within this list are processed when the sequencer is active. In this thread, the event type information is used to decide whether a region should be added, removed or left unchanged in the active region list.

Given the region based nature of this processing, the use of time-based effects (such as delays and reverbs) can be problematic with the decaying effect for a region potentially being cut off as the regions timing events dictate that it has finished playing. To overcome this, CwS uses im-

pulse responses to calculate the decay time of a given card featuring time based effects. This timing information is then used to extend the processing time of the given region to ensure that the decay is never cut off.

## 6. WORKSHOPS

The software package has been presented at numerous workshops across multiple countries in the European Union. Feedback on software tends to be overwhelmingly positive due to the streamlined user experience through working with cards. Participants and workshop leaders have noted that the software is excellent as an introductory tool for engaging sonic exploration and creativity for four key reasons.

Firstly, the card system enables people to clearly see and arrange the sounds they are working on without needing to use linguistic markers. Some workshop leaders stressed the importance of allowing students to work with drawn images of sounds. In this context, being able to have a card with an image enables a wide range of participants from the very young upwards to intuitively use the software.

Secondly, projects such as Sonic Postcards [7] alongside the work of Holland [23] present a strong case for the further development of student listening activities through field recording expeditions with groups of participants. The card system means that recordings from several groups of participants can be easily combined into a set of sounds for them to compose with.

Thirdly is the level/template system which many facilitators have responded positively to. By not presenting a long list of options to a user, participants can spend more time making music and less time choosing effects or processes. Teachers have welcomed the fact that in CwS, the children are gradually introduced to more tools as they work through the levels. CwS encourages users to explore each effect in turn and develop understanding before they move to the next level with a teacher in support.

Finally, commentators have noted the visual models within the system. Upon illustrating the software and similar tools in workshop contexts, participants are often drawn to similar starting points/processes. Delay (and echo), Pitch shifting (and harmonising), Reverse and Reverb are often the

“go-to” processes. Great care has been taken with the visual models for these processes. Participants have almost universally praised their clarity, reducing the need for lengthy explanations. Several practitioners have highlighted the reverb model being particularly helpful in visually linking the room size and the surface material of the room with sonic changes when using the controls. Such an approach is a noticeable contrast to software packages such as Audacity that is often used in schools by workshop leaders. All of these approaches subsequently encourage focused listening and developmental exploration and creativity with the tools available in the software.

## 7. FUTURE DEVELOPMENTS

While ongoing production and small enhancements for CwS are planned across Windows and Mac OS, two significant developments are currently underway under the umbrella of the Interfaces project. The first of these is the addition of an audio visual layer to the software that would allow users to sequence music to video. This opens up the system to a broader audience interested in sound for moving image and sound design. The second major development is the implementation of the Compose with Sounds Live (CwS Live) platform. This tool aims to expose students to the world of mixed media composition and performance via the transformation of live audio input alongside the triggering of recorded materials. The system is also designed to encourage collaborative performance. To do so, the software allows control data to be sent to other instances of the itself on a local area network via UDP.

## Acknowledgments

The authors would like to thank their respective institutions and the European Commission’s Creative Europe programme for supporting the ongoing development of the project. They would also like to thank Dr. David Moore and David Devaney for their contributions to the Free Sound Object Mixer and the CwS Version 1.0.

## 8. REFERENCES

- [1] L. Landy, *Understanding the Art of Sound Organization*. Mit Press, 2007.
- [2] S. Pearse, D. Moore, and D. Devaney, “Free sound object mixer,” <https://github.com/spearse/FSOM>, Aug 2015, accessed: 2019-03-07.
- [3] L. Landy, *Making Music with Sounds*. Routledge, 2012.
- [4] —, “Ears : Electroacoustic resource site,” <http://www.ears.dmu.ac.uk/>, accessed: 2019-03-07.
- [5] —, “Ears 2 : Electroacoustic resource site,” <http://ears2.dmu.ac.uk/>, accessed: 2019-03-07.
- [6] L. Landy, R. Hall, and M. Uwins, “Widening participation in electroacoustic music: The ears 2 pedagogical initiatives,” *Organised Sound*, vol. 18, no. 2, pp. 108–123, 2013.
- [7] Sound and Music, “Sonic postcards,” <http://www.soundandmusic.org/projects/sonic-postcards>, accessed: 2019-03-07.
- [8] “Interfaces network,” <http://www.interfacesnetwork.eu/>, accessed: 2019-03-07.
- [9] “The interfaces project,” <http://www.interfaces.dmu.ac.uk/>, accessed: 2019-03-07.
- [10] “wxwidgets,” <https://www.wxwidgets.org/>, accessed: 2019-03-07.
- [11] “Qt libraries & apis, tools and ide,” <https://www.qt.io/>, accessed: 2019-03-07.
- [12] K. Schwaber, “Scrum development process,” in *Business object design and implementation*. Springer, 1997, pp. 117–134.
- [13] P. Nascar Octavian, “Paulstretch: Paul’s extreme sound stretch,” <http://hypermammut.sourceforge.net/paulstretch/>, accessed: 2019-03-07.
- [14] C. Roads, *Microsound*. MIT press, 2004.
- [15] E. Gillet, “Mutable instruments — clouds,” <https://mutable-instruments.net/modules/clouds/>, accessed: 2019-03-07.
- [16] K. Maeda, “Performance evaluation of object serialization libraries in xml, json and binary formats,” in *Digital Information and Communication Technology and its Applications (DICTAP), 2012 Second International Conference on*. IEEE, 2012, pp. 177–182.
- [17] C. with Sounds Team, “Compose with sounds,” <http://www.cws.dmu.ac.uk/>, accessed: 2019-03-07.
- [18] Roli, “Juce,” <https://juce.com/>, accessed: 2019-03-07.
- [19] M. A. Boden, *The Creative Mind: Myths and Mechanisms*. Routledge, 2004.
- [20] T. Magnusson, “Designing constraints: Composing and performing with digital musical systems,” *Computer Music Journal*, vol. 34, no. 4, pp. 62–73, 2010.
- [21] D. Wood, J. S. Bruner, and G. Ross, “The role of tutoring in problem solving,” *Journal of child psychology and psychiatry*, vol. 17, no. 2, pp. 89–100, 1976.
- [22] J. Biggs, “Enhancing teaching through constructive alignment,” *Higher education*, vol. 32, no. 3, pp. 347–364, 1996.
- [23] D. Holland, “A constructivist approach for opening minds to sound-based music,” *Journal of Music, Technology & Education*, vol. 8, no. 1, pp. 23–39, 2015.

# INSIGHTS IN HABITS AND ATTITUDES REGARDING PROGRAMMING SOUND SYNTHESIZERS: A QUANTITATIVE STUDY

Gordan Kreković

Visage Technologies

gordan.krekovic@visagetechnologies.com

## ABSTRACT

Sound synthesis represents an indispensable tool for modern composers and performers, but achieving desired sonic results often requires a tedious manipulation of various numeric parameters. In order to facilitate this process, a number of possible approaches have been proposed, but without a systematic user research that could help researchers to articulate the problem and to make informed design decisions. The purpose of this study is to fill that gap and to investigate attitudes and habits of sound synthesizer users. The research was based on a questionnaire answered by 122 participants, which, beside the main questions about habits and attitudes, covered questions about their demographics, profession, educational background and experience in using sound synthesizers. The results were quantitatively analyzed in order to explore relations between all those dimensions. The main results suggest that the participants more often modify or create programs than they use existing presets or programs and that such habits do not depend on the participants' education, profession, or experience.

## 1. INTRODUCTION

During the last five decades, sound synthesis strongly contributed in shaping the path of music evolution. The technology that allowed creating an endless variety of novel sounds brought greater freedom in expressing musical ideas and encouraged musicians to be more innovative and ambitious in their artistic intentions. In order to increase flexibility of sound creation, synthesizers typically provide musicians with a large number of controllable parameters. However, since synthesis parameters do not necessarily bear acoustical meaning and they can depend on each other, managing numerical parameters is a difficult and time-consuming activity which can negatively affect inspiration and productivity [1].

In order to mitigate this problem, researchers have proposed solutions based on automatic selection of synthesis parameters which allow musicians to create desired sounds more intuitively. Instead of controlling numerical parameters manually, musicians can define their requirements in several other ways: (1) by providing a sound sample perceptually similar to the target sound, (2) by describing the target sound by using attributes (such as bright and harsh), and (3) by using more intuitive interfaces (such as visualizations of timbre spaces and scoring

of automatically generated sounds). Mapping those inputs or actions into synthesis parameters is a non-trivial problem that is usually approached using various computer science techniques.

Automatic selection of sound synthesis parameters is a relevant research challenge, especially nowadays when artificial intelligence is starting to emerge as a mean of advanced automation. Besides being academically interesting, automatic parameter selection has the potential to change the way how musicians use sound synthesizers. However, although this practical research topic is primarily motivated by possible pragmatic improvements, it has not been informed or guided by user experience (UX) studies. While the research has been ongoing for almost three decades, existing solutions are still not widely accepted in practical use and they are scattered across a variety of approaches and problem definitions.

A comprehensive study on attitudes and habits of musicians who use sound synthesizer might help in articulating the research question in terms of defining which specific problems automatic parameter selection should address. It may also help in explaining and assessing the relevance of different problem definitions and possible technical approaches. Thorough understanding of users' needs, habits, and attitudes will help researchers opt for design decisions which maximize usability and usefulness of their solutions. Insights about the practical context are also relevant for theoretical studies, which do not aim for applicability, but for demonstrating novel ideas and concepts, because the practical context provides the realistic expectations and allows the explicit ratio between theoretical knowledge and applicability. Finally, a study on users' habits and attitudes can also inform the process of designing new interfaces for sound synthesizers or at least serve as a starting point for further user research. Although a few UX studies related to sound synthesizers exist [2, 3] they are not fully aimed at informing the research on automatic parameters selection.

In light of that, the purpose of this paper is to investigate attitudes and habits of musicians who use software or hardware sound synthesizers. The approach is explorative and it also takes into account users' demographic data, profession, educational background and experience in using sound synthesizers, relating those dimensions to their habits and attitudes. To reach this objective, we conducted a questionnaire that included 122 participants and quantitatively analyzed results.

The paper starts with a brief overview of certain previous studies on automatic selection of synthesis parameters, which form an appropriate context of this research. The rest of the paper explains methodology, results, discussion, and conclusions of the quantitative research on programming sound synthesizers.

## 2. RELATED WORK

A considerable number of studies related to automatic selection of synthesis parameters emerged in the interdisciplinary field of computer music technology during the last three decades. Those studies explored three aforementioned ways of defining a desired sound: (1) by providing a perceptually similar sound, (2) by describing it using attributes, and (3) by interacting with a more intuitive user interface.

Matching a provided perceptually similar sound can be observed as an optimization problem in the space of synthesis parameters with the aim to minimize the perceptual difference between the provided sound and the synthesized result. Evolutionary algorithms are an appropriate approach to solving such kind of optimization problems, so they were the primary choice of many researchers, including the pioneers Andrew Horner and his colleagues. After reporting successful results with an FM synthesizer [4], they conducted similar studies for other sound synthesis techniques using the fitness function based on the same similarity measure – the absolute difference of two discrete Fourier transforms [5-7]. Some researchers noticed shortcomings of the selected similarity measure and proposed their solutions taking into account psychoacoustic phenomena [8-10]. Target matching using evolutionary algorithms with automatic calculation of the fitness function have been extensively studied and applied to various synthesis techniques including additive synthesis [11], subtractive synthesis [12], noise shaping [13], granular synthesis [14], plucked string synthesis [8], dynamic stochastic synthesis [15], and even synthesizers with multiple synthesis engines [16]. In addition to similarity measures calculated from the signal, several authors explored interactive evolutionary algorithms and proposed solutions that rely on the fitness values provided by the user [17-19].

Besides evolutionary algorithms, some studies explored and applied other computer science techniques such as fuzzy logic [20] and deep neural networks [21].

Marginally related to the problem of target matching are feature synthesizers capable of producing sounds from a given set of audio features that are either extracted from a target sound or provided by the user [22-26].

In contrast to automatic target matching, only several authors focused on controlling sound synthesizers using timbral attributes. Miranda presented a system based on decisions trees used to induce relations between quasi-timbral attributes and synthesis parameters [1], while Gounaropoulos and Johnson employed a neural network to learn relations between adjectives and audio features of a sound characterized by those adjectives [27]. Another approach is decomposing the inherently complex problem into two simpler steps: the first one is mapping timbral attributes into audio features using an expert system

based on fuzzy logic, while the second step is a pseudo-heuristic search for appropriate synthesis parameters to match the target audio features [28].

Some other notable solutions include: a knowledge-based system for controlling FM synthesizers [29], a system for sound synthesis and transformation based on adjectives, SeaWave [30], keyword analysis and clustering [31], an expert system for mapping adjectives directly to sound synthesis parameters [32], and an interactive evolutionary algorithm extended with adjective control [33].

In most of the aforementioned literature, the focus was on technical solutions without detailed explanations of the problem from the users' point of view. Moreover, it is rarely clear how the proposed solutions are intended to be used – as a tool that supports users when creating new sounds, as a tool that completely offloads users from that type of work, as a tool for inspiring musicians, or something else. For that reason, the aim of this paper is to strengthen the user dimension and draw more attention to user experience.

## 3. METHODOLOGY

Quantitative results presented in this paper are obtained by analyzing responses to a questionnaire about habits and attitudes of sound synthesizer users when creating, modifying, and using sounds. The questionnaire, which mostly consisted of questions with predefined ordinal and categorical answers, was divided into two sections.

The first section included the questions about participants' demographics (gender and age), primary field of work or education (since some synthesizer users might not be professional musicians or music students), level of music education, and experience with sound synthesizers (i.e. duration of use).

The second section dealt with participants' habits and attitudes regarding using, modifying, and creating programs in sound synthesizers. The questions from the second section were about (1) their tendencies to use predefined or existing programs, modifying existing programs, and creating new programs from scratch, (2) actions the participants are likely to take when the desired sound is not predefined, (3) impediments of creating and modifying programs manually, (4) features of sound synthesizers that can help them most in creating and modifying programs, and (5) potential helpfulness of hypothetical functions for automatic or semiautomatic selection of synthesis parameters. Most of the questions consisted of a common part (e.g. "How often do you take the following actions when using sound synthesizers?") and a specific statement (e.g. "Using predefined programs from the synthesizer without modifications", "Creating your own programs from scratch") treated as an ordinal question with a 5-point Likert scale (e.g. "never", "rarely", "sometimes", "very often", "always"). Besides the questions with predefined ordinal or categorical answers, there was also an optional question about the challenges that users face when creating or modifying programs. The whole questionnaire is available here:

<https://goo.gl/forms/3Tc7XBolKlJzr19k2>

When the questionnaire was ready and validated through test runs, it was disseminated using Facebook



groups, Internet forums, and direct contacts. One Facebook group was more oriented towards researchers, while all other groups and forums were general user groups not focused on any particular music creation tools, brand, or product of music industry.

The quantitative analysis of the collected responds included calculating statistics for each question, as well as analyzing pairs of answers. Appropriate statistical tests were selected based on answer types: Wilcoxon rank-sum test for pairs of categorical and ordinal answers, chi-squared hypothesis tests for pairs of categorical answers, and Spearman's rank correlation for correlations between ordinal answers.

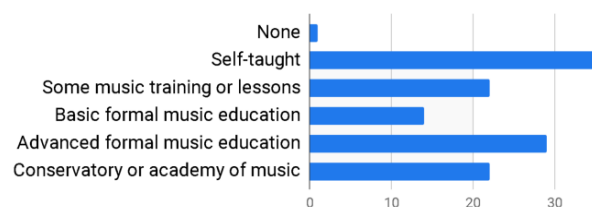
Such quantitative research methods are typically used in the field of human-computer interaction and the same methodology was applied to some topics related to music [34-36].

## 4. RESULTS

### 4.1 About Participants

The demographic structure of the participants suggests a significant gender bias with 95.1% of male participants and only 2.5% of female participants. Since the responses were collected using a non-discriminatory, unbiased, and anonymous method, this statistic may indicate a gender bias in the field, similar to disproportions found in other technical domains [37]. The mean age of participants is 41.3, while the standard deviation is 12.2.

As their primary field of professional work or formal education, most of the participants stated computers and technology (28.2%), music (28.2%), arts and communications (12.1%), and management, business and finance (9.7%). Regarding their educational background in music, one participant had no education in music, 28.2% are self-taught, 17.7% had some training or lessons, 11.3% have basic music education (elementary music school, preparatory school, etc.), 23.4% have an advanced formal education, while 17% completed a conservatory or academy (Bachelor or Masters of Music and higher) as shown in Figure 1. As expected, the Wilcoxon rank-sum test confirmed that participants who are professional musicians or music students have significantly higher levels of music education than the others ( $p < .01$ ).



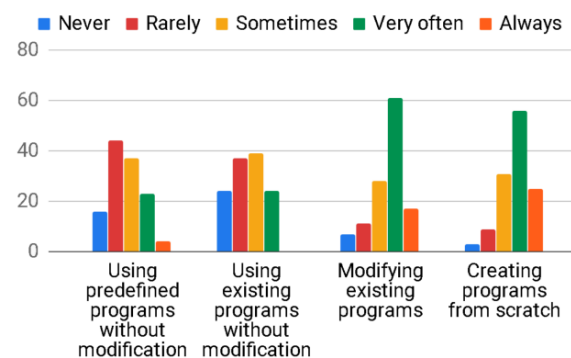
**Figure 1.** Participants' music education levels.

In general, the participants have a lot of experience with using sound synthesizers, since 71% of them have been using sound synthesizers for more than 10 years, 14.5% between three and ten years, 9.7% between one and three years, and 2.4% between three months and a year, while the remaining 2.4% are novice users with less than 3

months of experience. According to the Spearman's correlation coefficient, there is a very weak positive correlation between the participants' experience and the level of music education ( $r_s = .14$ ,  $p < .01$ ), and a moderate positive correlation between the experience and their age ( $r_s = .48$ ,  $p < .01$ ).

### 4.2 Usage Habits

In order to ascertain the usage habits, the participants were asked to state how often they take the following actions when using sound synthesizers: 1) using predefined programs (i.e. presets) without modifications (activity A1), 2) using existing programs created by others without modification (A2), 3) modifying predefined or existing programs (A3), and 4) creating programs from scratch (A4). For each activity, the possible answers were based on a Likert frequency scale. The distributions per action are shown in Figure 2. The results of the Wilcoxon rank-sum tests between all pairs of activities indicate that the participants more often modify existing programs (A3, Median=Very often) or create new programs from scratch (A4, Median=Very often) than they use presets (A1, Median=Sometimes) or existing programs without modification (A2, Median=Sometimes). All Wilcoxon rank-sum tests conducted between the pairs A1-A3, A1-A4, A2-A3, and A2-A4 confirmed statistically significant differences between answers ( $p < .01$  for all the aforementioned pairs), while no such differences were found between A1-A2 and A3-A4. These observations have been made considering all the participants as one group, but the same results have been obtained on specific subgroups: the participants with less than 3 years of experience with sound synthesizers, the participants who are not professional musicians or music students, and even the participants without formal music education (i.e. those without any education, self-taught participants, and those who had some trainings or lessons).



**Figure 2.** Frequencies of taking different actions when using sound synthesizers.

The indication that the aforementioned habits are neither related to the participants' experience nor their music education level has been confirmed by calculating Spearman's correlation coefficients between those dimensions. All the values of the obtained coefficients were between -0.1 and 0.1 with  $p > .5$ .

The next question in the survey also referred to the usage habits. The participants were asked how likely they

would take the following actions if they needed a sound that was not included in the presets: 1) use another synthesizer that might have such a program (action *B1*), 2) search for an appropriate program for their synthesizer (e.g. online) (*B2*), 3) modify one of the presets (*B3*), and 4) create their own program from scratch (*B4*). The results suggest that the participants are in general least likely to search for an appropriate program (*B2*, Median=*Unlikely*), undecided when it comes to using another sound synthesizer (*B1*, Median=*Undecided*), and likely to modify one of the presets (*B3*, Median=*Likely*) or create new programs from scratch (*B4*, Median=*Likely*). The series of Wilcoxon rank-sum tests between the pairs of answers confirmed that there are statistically significant differences between *B1-B2*, *B1-B3*, *B1-B4*, *B2-B3*, and *B3-B4* ( $p < .01$ , for all the pairs), while there is no significant difference between *B3-B4*. These findings have been made by considering all participants, but the same results have been obtained for the subgroup of the participants without formal music education and those who are not professional musicians or music students.

The Spearman's correlation coefficients show that the likelihood of taking aforementioned actions is neither monotonically correlated with the participants' experience nor their music education level, because all of the coefficient values were between  $-0.1$  and  $0.1$  with  $p > .5$ .

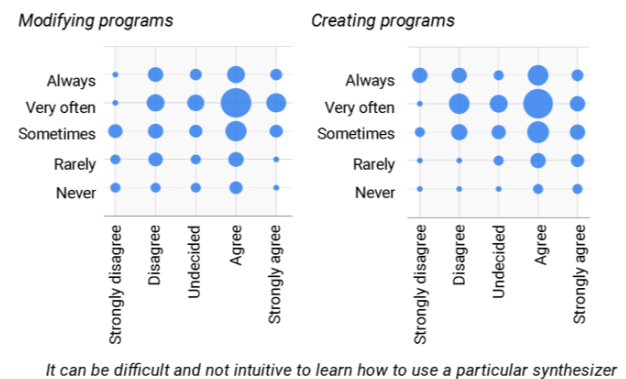
### 4.3 Impediments of Synthesizer Programming

Understanding impediments is as equally important as understanding usage habits. The participants were asked to express their level of agreement on a five-point Likert scale with the following statements about impediments of creating and modifying programs manually: 1) it can be time consuming (impediment *I1*), 2) it can distract them from focusing on music (*I2*), 3) it can be difficult and not intuitive to learn how to use a particular synthesizer (*I3*), and 4) it rarely leads to the desired results (*I4*). The participants in general agreed with the statements about the time consumption (*I1*, Median=*Agree*), distraction (*I2*, Median=*Agree*), and lack of intuitiveness (*I3*, Median=*Agree*), but disagreed with the last statement (*I4*, Median=*Disagree*). The same results have been obtained for all participants, but also for the specific subgroups: the participants with less than 3 years of experience, the participants who are not professional musicians or music students, and those without formal music education.

A weak negative monotonic correlation has been found between the statement *I4* and the participants' experience (Spearman's correlation:  $r_s = -0.30$ ,  $p < .01$ ) generally suggesting that the longer the participants use the synthesizers, the less they agree that manual programming rarely leads to the desired results. An even more interesting finding is a weak positive monotonic correlation between the education level and the statement *I3* (Spearman's correlation:  $r_s = 0.25$ ,  $p < .01$ ). The higher education participants have, the more they agree that it can be difficult and not intuitive to use a particular sound synthesizer.

Regarding the relations between habits and impediments, the participants who use presets without modifying them more often (*A1*), rated all the statements about impediments with generally higher points. The Spear-

man's correlation coefficients suggest weak positive monotonic correlations between the activity *A1* and the statements *I1* ( $r_s = .20$ ,  $p < .05$ ), *I2* ( $r_s = .26$ ,  $p < .01$ ), *I3* ( $r_s = .29$ ,  $p < .01$ ), and *I4* ( $r_s = .27$ ,  $p < .01$ ). On the other hand, the more often participants create programs from scratch, the less they consider the lack of intuitiveness (*I3*) and the risk of getting undesired results (*I4*) as impediments. The Spearman's correlation coefficients indicate weak negative correlations between the activity *A4* and statements *I3* ( $r_s = -0.22$ ,  $p < .01$ ) and *I4* ( $r_s = -0.28$ ,  $p < .01$ ). Figure 3 illustrates the mentioned relations between the pairs *I3-A3* and *I3-A4*.



**Figure 3.** Left: relation between the statement about difficulty of synthesizer programming and the frequency of modifying existing programs. Right: relation between the statement about difficulty and the frequency of creating programs from scratch.

The participants could optionally answer an open-ended question to expand on other challenges they face when creating and modifying programs manually. Out of 122 participants, 34 of them decided to take the opportunity and share their opinion. Most of the answers can be organized in four main groups: 1) challenges related to user interfaces (11 answers), 2) challenges related to learning and understanding the synthesis process (7 answers), 3) challenges related to limited or missing features of specific sound synthesizers (7 answers), and 4) challenges rooted in the creative process (5 answers).

Inefficient user interfaces were most frequently mentioned in the participants' comments. Some of them focused on problems with deep menus (e.g. "Straightforward vs menu-diver interfaces" and "Menu diving. Wish more manufacturers would surface more of their controls."), while the others criticized inconsistency (e.g. "Some knobs are named different for the same effect" and "Thinking more of VI synths - there is so much inconsistency in the UI design that much time is lost understanding what the devs actually want you to do. In contrast, physical synths often (though certainly not always) offered a clearer view of the signal path, simply by their physical layout.").

The participants also identified a lot of challenges related to learning, especially due to diversity among sound synthesizers (e.g. "All synths are so different in character, knobs, etc, it takes time to get used to them" and "Different sorts of synthesis require different background

knowledge, most of which have steep learning curves that are at least partially exclusive. In other words, there is an enormous investment of time to deeply learn how the different forms of synthesis work. This learning is a prerequisite to effective use of synthesizers.”).

A significant number of comments touched upon limitations and lacking features in sound synthesizers. Some examples are: “Running into ‘dead ends’, i.e. discovering that a certain function or effect is needed to achieve the desired result, e.g. delay or an extra LFO, or settings not reaching far enough.”, “Usually only limitations of that synth, or polyphonic Vs monophonic, number of oscillators”, and “The limitation of the synthesiser, in that they all have their own ‘sound’ (as generally defined by it’s Oscillators and Filters) and so if you’re aiming for a result on the edge of that ‘sound’ then you can get close to or (worse) hit the limits of that synth”.

Finally, the comments about the creative process were not directly related to sound synthesizers, but opened interesting concerns highly relevant in the context of synthesizer programming, e.g.: “Building the acoustic landscape across multiple patches”, “Having a listen to inspirations and being 100% unmotivated to even try”, and “If working with other musicians, who aren’t present, your sound cannot be considered complete until you’ve played it in context with the other parts”.

#### 4.4 Facilitating Synthesizer Programming

The last part of the questionnaire focused on aspects that help users in creating and modifying programs manually. The first question in that section had a categorical list of all improvements from which the participants could choose exactly one that could help them most, or write their own answer in the “Other” category. The participants mainly opted for intuitive user interfaces (58.1%), informative guides on how to use the synthesizer such as manuals, tutorials, and online material (25.8%), and excellent presets that can inspire users or serve as a starting point for modification (11.3%).

The participants who selected one of those three most frequent answers have been divided in three groups based on their answers. Usage habits between those groups were compared using a set of Wilcoxon rank-sum tests. The results indicate that the participants who think that the user interface can help them most create new programs more often ( $p < .01$ ), but modify existing programs less often than the participants who think that excellent presets can help them most ( $p < .05$ ). Other statistically significant differences have not been found.

In the last question, the participants were asked to rate potential helpfulness of the following functions in creating synthesizer programs: 1) the user chooses a category and the system generates new, random programs that fit the category ( $F1$ ), 2) The user describes a desired sound using attributes (e.g. bright and percussive) and the system generates such a program ( $F2$ ), 3) the user provides an audio sample and the system generates a program that sounds similarly ( $F3$ ), and 4) the user manipulates the graphical interpretation of the sound using an intuitive GUI and the system modifies the program appropriately ( $F4$ ). The results of the Wilcoxon rank-sum tests between

all pairs of activities indicate that the participants consider functions  $F3$  (Median=Helpful) and  $F4$  (Median=Helpful) more helpful than functions  $F1$  (Median=Slightly helpful) and  $F2$  (Median=Slightly helpful). All the Wilcoxon rank-sum tests conducted between the following pairs:  $F1$ - $F3$ ,  $F1$ - $F4$ ,  $F2$ - $F3$ ,  $F2$ - $F4$  confirmed statistically significant differences between the answers ( $p < .05$  for all the aforementioned pairs), while no such differences were found between  $F1$ - $F2$  and  $F3$ - $F4$ . These findings have been made considering all participants, but the same results have been obtained for the participants without formal music education and those who are not professional musicians or music students.

The participants with a higher music education might consider using attributes ( $F2$ ) more helpful, as indicated by a weak positive Spearman’s correlation coefficient ( $r_s = 0.23$ ,  $p < .05$ ). No monotonic correlation has been found between the participants’ usage experience and the helpfulness of the proposed functions. Still, usage habits seem to be related to the perception of helpfulness: the more often participants modify existing programs, the more helpful they consider all the functions (Spearman’s correlation for  $F1$ :  $r_s = 0.24$ ,  $p < .01$ , for  $F2$ :  $r_s = 0.22$ ,  $p < .05$ , for  $F3$ :  $r_s = 0.29$ ,  $p < .01$ , and for  $F4$ :  $r_s = 0.15$ ,  $p < .01$ ). On the other hand, the more often the participants create new programs from scratch, the less helpful they consider function  $F2$  (Spearman’s correlation:  $r_s = -0.22$ ,  $p < .05$ ).

## 5. DISCUSSION

Before discussing the results, this section starts with several topics regarding the methodology and scope that are important for interpreting the results.

Since the study primarily relies on the quantitative survey methodology, some known potential biases can affect the results. The questionnaire was carefully designed to minimize those biases: the questions were formulated showing a neutral stance toward different answers, while the terminology was selected and refined during test runs to be as accurate as possible. Also, the introductory text emphasized that the goal of this independent research is to better understand attitudes and habits regarding programming sound synthesizers. However, one particular type of biases, which could have appeared in this research, was not fully controllable by the survey design and the selection of participants. It is the social desirability bias. The questionnaire was disseminated in multiple groups on social networks and online forums that gather synthesizer enthusiasts, hobbyists and professional practitioners, and even researchers in the field of computer music technology. Deep exploration of sound synthesizers, manual synthesizer programming and tweaking synthesis parameters are probably considered as highly respected activities within some of those groups. Even though there is no clear evidence that this fact affected the questionnaire results, the social desirability bias and post-rationalization represent possible risks for quantitative data regarding usage habits. To explore these risks further and mitigate them in future studies, different research methodologies can be used such as diary/camera studies or unmoderated user experience studies.



Another important observation is that habits may be related with purposes of using sound synthesizers (e.g. studio recording vs. live performance, playing different instruments vs. experimenting with sounds, different music genres, etc.). Additionally, as mentioned by one of the participants in the Facebook comments, usage habits may be different for different types of sound synthesizers, especially because of differences between hardware and software synthesizers that might have various concepts of user interfaces and incomparable levels of affordability. Although the questionnaire was designed to cover multiple dimensions, the questions about usage purposes and types of sound synthesizers were not included. The reason is that those topics would require multiple additional questions, as participants may use various types of synthesizers for various purposes. Such extensions of the questionnaire would significantly increase the complexity of analysis and broaden the scope of the study, possibly removing the focus from the current research questions. However, that does not mean that purposes and types are not important dimensions. Understanding users' habits and attitudes in relation to purposes and types of sound synthesizers may be very valuable insights for making comprehensive conclusions. Now, when this study has shown that the music education and the usage experience do not have a significant impact on the usage habits, future research can be more focused on purposes and synthesizer types.

One of the most notable finding in this research is the fact that the questionnaire participants more often modify or create programs manually than they use presets or programs created by others. This is especially interesting because apparently such habits do not depend on music education or experience in using sound synthesizer. A possible concern is that this conclusion may be specific to the group of participants involved in this study. If the group contained more keyboard players who prefer using imitative sounds, the percentage of participants who often rely on existing programs would probably be higher. However, since there were 122 participants acquired from multiple online forums and Facebook groups, the group size and the acquisition procedure should have mitigated a potentially strong sampling bias. Considering the number of participants and their experience in using sound synthesizers, it is valuable to quantitatively analyze habits and attitudes habits of such users, and the results are at least indicative. The synthesizer users similar to the survey participants seem to enjoy the process of creating novel and authentic sounds. The less experienced participants more often acknowledge the fear of getting undesired results by manual synthesizer programming, but that does not seem to demotivate them, as their habits are same as the habits of more experienced users.

Another consistent conclusion is related to user interfaces. A significant number of the participants stated that better user interfaces could help them most in synthesizer programming. They also mentioned various specific problems with user interfaces within their open-ended responses. Knowing that the users generally modify or create programs quite often, user interfaces are inevitably the crucial medium between the user and the synthesis engine, strongly influencing the perception, expectations,

and general experience with synthesizer programming. This seems to be recognized by manufacturers of hardware and developers of software synthesizers, as layouts with lots of direct controllers have restored their popularity during the last decade. Another recent trend are hardware devices – so called synthesizer programmers – that can be attached to synthesizers in order to extend their user interfaces. Together with the results of this study, the recent trends provide evidence about the importance of user interfaces. For that reason, user research practices should have a very high priority among those research activities aimed at improving user experience with synthesizer programming. All pragmatically-oriented and technical solutions should be grounded on the UX studies, but this is not the case at the moment.

The open-ended response revealed one interesting point that was not covered by predefined questions in the survey. Some of the participants mentioned limitations or missing features in sound synthesizers as a problem that reflects on synthesizer programming. Therefore, it seems that improving user interfaces may not be sufficient to improve the general user experience. Evidently, some users feel that they sometimes cannot achieve desired sounds, not because of inefficient user interfaces or their lack of knowledge, but because of characteristics of underlying synthesis engines. User interfaces together with synthesis engines have an inseparable effect on sound creation, so both parts should be designed by following informed choices based on UX research.

This study has also shown what the participants think about functionalities for automatic selection of synthesis parameters. The corresponding question was deliberately formed to cover the main approaches explored the previous work in this field. Since the participants expressed more hope in potential helpfulness of target sound matching and GUI-based methods, the results should be interpreted carefully, as the participants did not have an opportunity to try those functions in practice or learn more about them, so they could have had very different ideas about the mentioned functions. For that reason, the future research direction should not be based on this single question, but the results are again indicative, especially the fact that the participants, who modify programs more often, consider all of those functions more potentially helpful. It is generally encouraging to see a positive or at least neutral attitude toward such novel and non-standard approaches.

With other results taken into consideration, it seems that supportive technology should only partially facilitate synthesizer programming, and not fully take control. The participants like to modify and create programs, and technology should help and inspire them, not hinder their creative engagement. In practical sense, that would mean introducing more interactive possibilities [15-17, 31] or generating multiple programs that users can selectively apply for further modifications. The latter concept can be inherently supported by all of those algorithms that rank potential programs and then present only the best one as a result. Examples are solutions based on genetic algorithms that can be easily extended to present multiple programs to users.



## 6. CONCLUSION

The conclusions outlined in this section are based on the quantitative results and the subsequent discussion, except the first one that emerges from the literature review. The presented conclusions can serve as inputs for synthesizer design, future studies on automatic selection of synthesis parameters, and future user research in the field of sound synthesis.

The first conclusion, based on the literature review, concerns the observed lack of user research in the existing solutions for automatic selection of synthesis parameters. While technical solutions employ advanced computer science techniques to resolve the problem of synthesizer programming, there is no evidence that the problem is appropriately formulated. Some of the previous studies conducted user testing, but only to demonstrate that solutions work well. The missing part is an investigation whether the solution would be more usable if it was based on a different approach. User experience studies should serve as one of information sources when deciding upon the solution's architecture and its argumentation in scientific publications.

The second conclusion is one of the most important quantitative results of this study. It is the fact that the participants more often modify or create programs manually than the use existing presets and programs. This result can influence the future direction of developing solutions for automatic parameter selection that target users similar to the participants of this research. Instead of aiming at synthesizing final sounds, those solutions could be designed to efficiently support the synthesizer programming process that enthusiastic users apparently prefer over using existing programs.

The existing and missing correlations in the result suggest that the habits regarding synthesizer programming are not related to user's music education or experience, but on the other hand, they are related to users' perception of impediments and helpfulness of possible solutions. For example, the participants who modify existing programs more often, agreed more with all the impediments and also considered all proposed functions more potentially helpful, but that was not the case with the users who create programs more often. Of course, the correlations do not confirm causalities and it is not possible to conclude whether habits form a perception of impediments, impediments form habits, or those dimensions are not causally related at all. However, the correlations are a very important reminder that not all users are the same and that particular solutions should aim to satisfy specific needs and expectations. When designing a novel solution for automating selection of synthesis parameters, a starting point should be based on the intended purpose and target users.

Finally, as a general remark regarding possible solutions for more efficient synthesizer programming, the results of this study show that the participants believe that the most helpful improvements would be those in user interfaces. While this result may be affected by the fact that users perceive the sound synthesis technology and its possibilities through user interfaces and thereby assign all problems and potential solutions to the interface level,

this is still an interesting insight, especially for practically-oriented solutions. Improvements of user interfaces or interactive approaches to automatic synthesis parameters selection are surely not the only mean of facilitating synthesizer programming, but they may be a safe starting point. Although the results of this research may provide some general guidelines for user interface design and overall solution conceptualization, they are not sufficient to inform all design decisions, as their purpose was to provide insights in habits and attitudes regarding synthesis programming and not to answer specific questions. Therefore, the design process should be informed by a carefully conducted user research based on the appropriate methodology.

## 7. REFERENCES

- [1] E. Miranda, "An artificial intelligence approach to sound design," in *Computer Music Journal*, vol. 19, no. 2, 1995, pp. 59–75.
- [2] A. Seago, S. Holland, and P. Mulholland, "A Critical analysis of synthesizer user interfaces for timbre," in *Proceedings of the XVIII British HCI Group Annual Conference*, UK, 2004.
- [3] C. Rasmussen, "Evaluating the Usability of Software Synthesizers: An Analysis and First Approach", MSc Thesis, University of Guelph, 2018
- [4] A. Horner, J. Beauchamp, and L. Haken, "Machine tongues XVI: Genetic algorithms and their application to FM matching synthesis," in *Computer Music Journal*, Vol. 17, No. 4, 1993, pp. 17–29.
- [5] A. Horner, "Wavetable Matching Synthesis of Dynamic Instruments with Genetic Algorithms," in *Journal of the Audio Engineering Society*, Vol. 43, No. 11, 1995, pp. 916–931.
- [6] A. Horner, "Auto-Programmable FM and Wavetable Synthesizers," in *Contemporary Music Review*, Vol. 22, No. 3, 2003, p. 21–29.
- [7] S. Wun and A. Horner, "A Comparison Between Local Search and Genetic Algorithm Methods for Wavetable Matching," in *Journal of the Audio Engineering Society*, Vol. 53, No. 4, 2005, pp. 314–325.
- [8] J. Riionheimo and V. Välimäki, "Parameter Estimation of a Plucked String Synthesis Model Using a Genetic Algorithm with Perceptual Fitness Calculation," in *EURASIP Journal on Applied Signal Processing*, Vol. 8, 2003, pp. 791–805.
- [9] M. J. Yee-King and M. Roth, "SynthBot – An Unsupervised Software Synthesizer Programmer," in *Proceedings of the International Computer Music Conference*, Ireland, 2008.
- [10] J. McDermott, "Evolutionary Computation Applied to the Control of Sound Synthesis", PhD Thesis, University of Limerick, Ireland, 2008.

- [11] A. Horner, "Envelope Matching with Genetic Algorithms," *Journal of New Music Research*, Vol. 24, No. 4, 1995, pp. 318–341.
- [12] M. J. Yee-King, "The Evolving Drum Machine", *Music-AL workshop, ECAL Conference*, 2007
- [13] M. Chinen and N. Osaka, "Genesynth: Noise Band-Based Genetic Algorithm Analysis/Synthesis Framework," in *Proceedings of the International Computer Music Conference*, Denmark, 2007
- [14] I. Fujinaga and J. Vantomme, "Genetic Algorithms as a Method for Granular Synthesis Regulation," in *Proceedings of the International Computer Music Conference*, Denmark, 1994, pp. 138–138.
- [15] J. Young, "Rethinking Synthesis: Extending and Exploring Gendyn," *BA Thesis*, University of Sussex, UK, 2010
- [16] M. Macret, "Automatic tuning of the OP-1 synthesizer using a multi-objective genetic algorithm," in *Proceedings of the Sound and Music Conference*, 2013, pp. 614–621.
- [17] C. Johnson, "Exploring Sound-Space with Interactive Genetic Algorithms," in *Leonardo*, Vol. 36, No. 1, 2003, pp. 51–54.
- [18] P. Dahlstedt, "A MutaSynth in Parameter Space: Interactive Composition through Evolution", *Organised Sound*, Vol. 6, No. 2, 2001, pp. 121–124.
- [19] J. Mandelis, and P. Husbands, "Musical Interaction with Artificial Life Forms: Sound Synthesis and Performance Mappings," in *Contemporary Music Review*, Vol. 22, No. 3, 2003, pp. 69–77.
- [20] B. Hamandicharef and E. Ifeachor, "Intelligent and Perceptual-Based Approach to Musical Instruments Sound Design," in *Expert Systems and Applications*, Vol. 39, No. 7, 2012, pp 6476–6484.
- [21] M. J. Yee-King, L. Fedden, and M. d'Inverno, "Automatic Programming of VST Sound Synthesizers Using Deep Networks and Other Techniques," in *IEEE Transactions on Emerging Topics in Computational Intelligence*, Vol. 2, No. 2, 2018, pp. 150–159.
- [22] D. Wessel, "Timbre Space as a Musical Control Structure," in *Computer Music Journal*, Vol. 3, No. 2, 1979, pp. 45–52.
- [23] C. Hourdin, G. Charbonneau, and T. Moussa, "A Sound Synthesis Technique Based on Multidimensional Scaling of Spectra," in *Computer Music Journal*, Vol. 21, No. 2, 1997, pp. 40–55.
- [24] T. Jehan, "Perceptual Synthesis Engine: An Audio-Driven Timbre Generator," *MSc Thesis*, Massachusetts Institute of Technology, 2001.
- [25] J. W. Beauchamp, *The Sound of Music: Analysis, Synthesis, and Perception of Musical Sounds*, Springer, New York, 2007.
- [26] D. Mintz, "Towards Timbral Synthesis: A New Method for Synthesizing Sound Based on Timbre Description Schemes," *Master thesis*, University of California, Santa Barbara, 2007.
- [27] A. Gounaropoulos and C. Johnson, "Synthesising Timbres and Timbre Changes from Adjectives/Adverbs," in P. Collet et al. *Applications of Evolutionary Computing*, Springer-Verlag, Berlin, 2006.
- [28] G. Kreković, A. Pošćić, D. Petrinović, "An Algorithm for Controlling Arbitrary Sound Synthesizers Using Adjectives," in *Journal of New Music Research*, Vol. 4, No. 45, 2016, pp. 375–390.
- [29] R. Ashley, "A Knowledge-Based Approach to Assistance in Timbral Design," in *Proceedings of the International Computer Music Conference*, The Hague, Netherlands, 1986.
- [30] R. Ethington and B. Punch, "SeaWave: A System for Musical Timbre Description," in *Computer Music Journal*, Vol. 18, No. 1, 1994, pp. 30–39.
- [31] R. Clement, "Automatic Synthesiser Programming," in *Proceedings of the International Computer Music Conference*, University of Huddersfield, UK, 2011, pp. 155–158
- [32] A. Pošćić and G. Kreković, "Controlling a Sound Synthesizer Using Timbral Attributes," in *Proceedings of the Sound and Music Computing Conference*, Stockholm, Sweden, 2013, pp. 467–472.
- [33] G. Kreković and D. Petrinović, "Intelligent Exploration of Sound Spaces Using Decision Trees and Evolutionary Approach," in *Proceedings of the International Computer Music Conference joint with Sound and Music Computing Conference*, Athens, Greece, 2014, pp. 1263–1270.
- [34] A. Pošćić, G. Kreković, and A. Butković, "Desirable Aspects of Visual Programming Languages for Different Applications in Music Creation," in *Proc. Int. Conf. Sound and Music Computing (SMC2015)*, Maynooth, 2015, pp. 329–336.
- [35] A. Pošćić and G. Kreković, "The Frailty of Formal Education: Visual Paradigms and Music Creation," in *Proceedings of the Audio Mostly*, London, 2017
- [36] A. Pošćić and G. Kreković, "Ecosystems of Visual Programming Languages for Music Creation: A Quantitative Study", in *Journal of the Audio Engineering Society*, Vol. 66, No. 6, 2018, pp. 486–494.
- [37] V. Armstrong, "Hard bargaining on the hard drive: gender bias in the music technology classroom," *Gender and Education*, Vol. 20, No. 4, 2008

# EXPERIMENTAL VERIFICATION OF DISPERSIVE WAVE PROPAGATION ON GUITAR STRINGS

**Dmitri Kartofelev**

Tallinn University of Technology,  
School of Science,  
Department of Cybernetics,  
Tallinn, Estonia  
dima@ioc.ee

**Joann Gustav Arro**

Tallinn University of Technology,  
School of Science,  
Department of Cybernetics,  
Tallinn, Estonia  
joann.arro@taltech.ee

**Vesa Välimäki**

Aalto University,  
School of Electrical Engineering,  
Department of Signal Processing and  
Acoustics, Acoustics Lab,  
Espoo, Finland  
vesa.valimaki@aalto.fi

## ABSTRACT

Experimental research into the fundamental acoustic aspects of musical instruments and other sound generating devices is an important part of the history of musical acoustics and of physics in general. This paper presented experimental proof of dispersive wave propagation on metal guitar strings. The high resolution experimental data of string displacement are gathered using video-kymographic high-speed imaging of the vibrating string. The experimental data are indirectly compared against a dispersive Euler-Bernoulli type model described by a PDE. In order to detect the minor wave features associated with the dispersion and distinguish them from other effects present, such as frequency-dependent dissipation, a second model lacking the dispersive (stiffness) term is used. Unsurprisingly, the dispersive effects are shown to be minor but definitively present. The results and methods presented here in general should find application in string instrument acoustics.

## 1. INTRODUCTION

Modern acoustic guitar strings are made of different metal alloys. Metal string produces a different sound to nylon or gut strings. Guitar strings may be “plain”, consisting only of a single material, like steel, nylon, or gut. Also they may be wound, having a core of one material and an overwinding of another.

Usually, plain guitar strings are not associated with significant dispersive effects like e.g. the bulkier steel piano strings are [1]. Thin guitar strings have a relatively low bending stiffness. The aim of this paper is to experimentally investigate and prove the possibility of transverse dispersive wave propagation on the second and third strings used in acoustic guitars.

Several models of transverse wave propagation on a stiff string, of varying degrees of complexity, have appeared in the literature [2–6]. Models with great emphasis on realistic frequency-dependence loss profiles are [7, 8]. These

models, intended for the synthesis of musical tones, are always framed in terms of a partial differential equation (PDE), or a system of PDEs. The simplified starting point for such models is the one-dimensional wave equation [9]. More realistic features, such as dispersion, various nonlinearities and frequency-dependent losses, are incorporated through several extra terms. Discussion in this paper is informed by the model equation proposed by Bensa *et al.* in [6], i.e., Euler-Bernoulli.

Some direct measurements of string vibration have been previously conducted. The methods of string displacement measurement can be roughly divided into three categories: the electromagnetic methods, electric field sensing, and optical methods. The electromagnetic methods exploit Faraday’s law, and the principle of the string displacement detection is the following: An electromagnetic coil is placed near the string, and the motion of the string induces a voltage in the circuit that is proportional to the string’s velocity from which the displacement of the string is obtained. This method was used and described in [10].

The electric field sensing makes use of the phenomenon of capacitance change between two electrodes, when the distance between them is varied. In the simplest approach, a conducting string is grounded, and direct current (DC) voltage is applied to an electrode plate. The string’s movement modulates the voltage between the string and the plate, and the information about the string’s displacement is obtained *cf.* [11].

The optical methods exploit various light or laser emitting and detecting sensors to capture vibration. For example, high speed cameras with suitable video analysis have been used to measure string vibration successfully [12]. Also, different devices that convert laser light into a uniform parallel beam and detect their shadows can ensure the result [13]. Devices that are based on various photovoltaic detectors have also been successful [14].

Our experimental approach can be classified under the aforementioned optical methods. We use a non-invasive video-kymographic method based on the exploitation of a digital high-speed line-scan camera (LSC) imaging. The method has been used successfully in musical acoustics research [15–17]. A monochord equipped with a guitar string is measured<sup>1</sup>. Experimentally obtained string vibration

<sup>1</sup> String set: Earthwood Light 2148. String gauge: 0.015 (15, 1115).

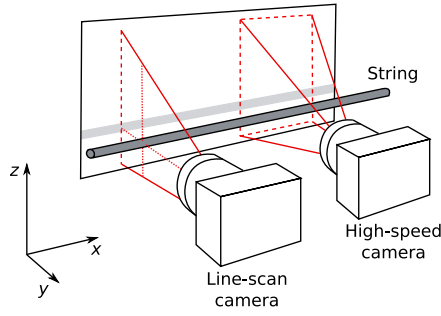


Figure 1. Operating principle of the LSC compared to an ordinary high-speed camera. Geometry of the area that is being imaged is shown on the white screen with the dashed red lines. Placement of the LSC with respect to the string vibrating in the  $z$ -direction while recording the vibration.

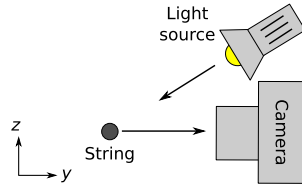


Figure 2. LSC recording string displacement. Cross-section of the recorded string is shown with the grey bullet.

data are then compared directly or indirectly against theoretical models, expanded upon below, with the aim to deduce some beneficial observations and reach conclusions.

Organisation of the paper is the following: Sec. 2 explains the experimental approach and set-up; Sec. 3 presents the dispersion analysis of the dispersive Euler-Bernoulli type model [6]. Numerically integrated solution of this model is presented and compared against its dispersion analysis; Sec. 4 presents a simpler time-stepping model of lossy non-dispersive string vibration, that is used here to identify dispersive features present in experimental data, and to distinguish them from other effects, such as frequency-dependent dissipation; Sec. 5 presents the experimental results and compares them against our assumptions and presented theory (the simplified model). Analysis and discussion of the results is directly informed by the Euler-Bernoulli type model; Sec. 6 concludes the paper.

## 2. EXPERIMENTAL MEASUREMENTS

The string displacement is measured using a LSC. The camera produces two-dimensional digital images (not videos) called the *kymographs*. The geometry of a digital imaging sensor of the LSC differs from a commonly used video camera. Usually, the video camera sensor pixels are placed in rows and columns forming a grid. The LSC sensor consists only of a single pixel array, referred here to, as the *line*, see Fig. 1. While filming the camera continuously stacks these lines to form an image. In addition, the global shutter technology allows for all pixels in a line to work

Manufacturer: Ernie Ball Inc. Coachella, California, USA 92236

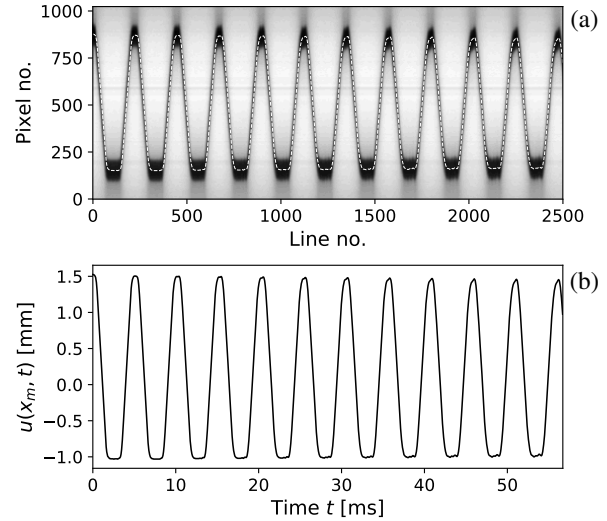


Figure 3. (a) Kymograph of vibrating string. Image recorded at 44 100 lines/s. String displacement  $u(x_m, t)$  tracking with line convolution method (2) is shown with the overlaid dashed line. (b) Calibrated string displacement time-series corresponding to the kymograph above.

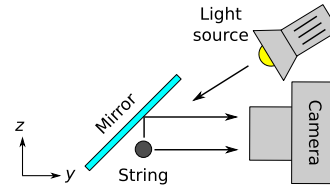


Figure 4. Dual polarisation measurement set-up [15].

as one (collect light simultaneously), preventing any image distortions to influence the recordings. Figures 1 and 2 show the perpendicular placement of the LSC with respect to the string while recording.

The string displacement time-series extraction from a kymograph is based on the discrete one-dimensional convolution integral of the individual kymograph lines

$$c[i] = (p * k)[i] = \sum_{n=-\infty}^{\infty} p[n] k[i - n], \quad (1)$$

where  $i \in [1, 1024]$  is the pixel number in any given line,  $p[i]$  is the image depth or colour value in bits, and  $k[i]$  is the convolution kernel — the image feature we are interested in. The kernel is selected to be roughly similar in shape to the string (its image), this guarantees that the convolved line  $c[i]$  will have a clear maximum (or minimum) that will coincide with the string position [16]. Thus, for any given line the pixel corresponding to string position

$$i = \arg \max c[i] \quad (\text{or } i = \arg \min c[i]). \quad (2)$$

This procedure is repeated for all kymograph lines. Figure 3a shows an example kymograph and the result of the image analysis. Figure 3b shows the calibrated time-series where the line number is multiplied by  $dt = 1/44\,100$  s since the camera is recording at audio sampling rate of



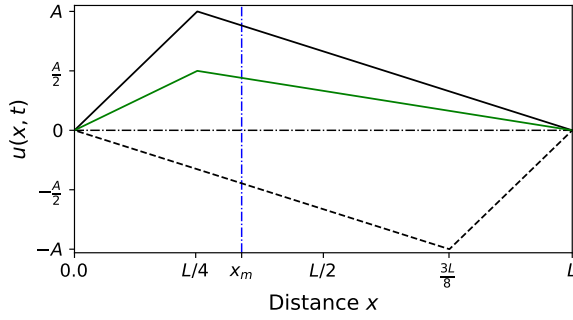


Figure 5. Schematic of the problem studied. Triangular shaped initial condition excited at  $x = x_e = L/4$  is shown with the solid black line, and the corresponding traveling waves (overlapping) are shown with the green line. The dashed line shows the string displacement at one half of the period. Vertical dash-dotted line shows the measurement coordinate  $x = x_m$  used in the experiment.

44 100 lines/s, and the pixel number  $i$  is multiplied by  $dx$  which value is determined by filming an object (high-contrast calibration sheet) with known dimensions.

## 2.1 Proof of planar vibration

In case a dual-polarisation measurement is required the LSC is used in combination with a mirror. The mirror is placed behind the string under a  $45^\circ$  angle with respect to the optical axis of the camera, as shown in Fig. 4. One half of the kymograph will contain displacement data for the vertical  $z$ -axis, and the other half for the horizontal  $y$ -axis.

The following method of controlled and repeatable string excitation is used in this study. The method is based on the fact that a thin cotton thread, when under great tensile load, snaps quite rapidly when heated abruptly (burned with a flame). The thread is looped around the string at the desired excitation point  $x = x_e$  along the string's speaking length, the string is then displaced to a suitable initial amplitude  $A$  in the desired direction with respect to the LSC. This procedure creates a triangular shaped initial condition shown in Fig. 5. Figure 6 shows that the guitar string excited in such a manner is capable of sustained planar vibration. At least for *some* time after the excitation.

## 3. DISPERSIVE STRING MODEL

The planar transverse vibration of a lossy stiff string can be described by

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} - \gamma^2 \frac{\partial^4 u}{\partial x^4} - 2\alpha \frac{\partial u}{\partial t} + 2\beta \frac{\partial^3 u}{\partial x^2 \partial t}, \quad (3)$$

where  $u(x, t)$  is the string displacement in  $z$ -direction and the non-negative  $\alpha$ ,  $\beta$ ,  $\gamma$  and  $c$  are the system parameter. The first term on the right-hand side of the equation, in the absence of the others, gives rise to ideal wave vibration, with traveling wave speeds  $c$  [9]. The second term introduces dispersion and is responsible for frequency-dependent wave velocity where constant  $\gamma$  is proportional to the bending stiffness. Last two terms allow for losses, and if  $\beta \neq 0$ , decay rates are frequency-dependant.

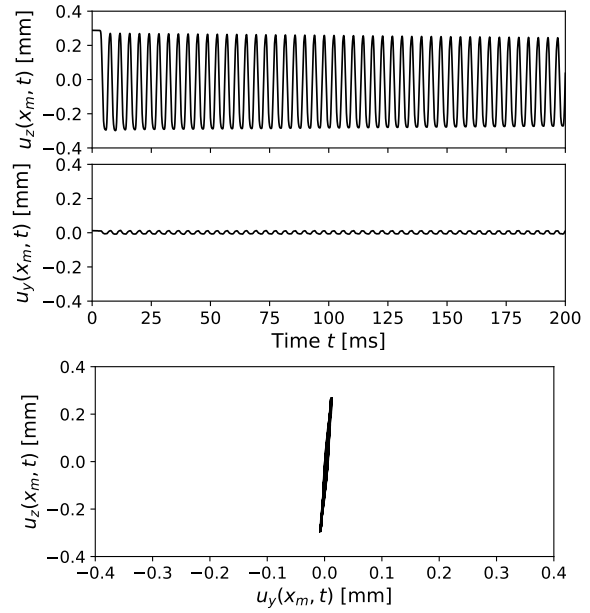


Figure 6. Persistent planar vibration. String displacement  $u$  is recorded at  $x = x_m$  using the measurement set-up shown in Fig. 4. Subscripts  $y$  and  $z$  indicate the direction of vibration as shown in Figs. 1 and 4.

## 3.1 Dispersion analysis and characteristic equation

The solution to Eq. (3) is assumed in the form

$$u \simeq u_0 e^{\varsigma t + i\kappa x}, \quad (4)$$

where the complex frequency  $\varsigma = \varsigma(\kappa)$  is a function of wavenumber  $\kappa$  and  $\kappa \in \mathbb{R}^+$ ,  $i$  is the imaginary unit, and  $u_0$  is an initial amplitude. Solving the characteristic equation

$$\varsigma^2 + 2q(\kappa)\varsigma + r(\kappa) = 0, \quad (5)$$

where

$$q(\kappa) = \beta\kappa^2 + \alpha, \quad r(\kappa) = c^2\kappa^2 + \gamma^2\kappa^4, \quad (6)$$

for  $\varsigma$  gives

$$\varsigma_{\pm} = -q \pm \sqrt{q^2 - r}. \quad (7)$$

These roots determine the behaviour of the general solution (4) of Eq. (3). Condition that the initial value problem corresponding to Eq. (3) be well posed is that roots (7) have real parts which are bounded from above as a function of  $\kappa$ ; this is to say that solution growth can be no faster than exponential, see assumption (4). Another physically relevant condition is that roots (7) have non-positive real parts for all  $\kappa$ , so that all exponential solutions are non-increasing (infinite string displacement). This condition is satisfied because  $q(\kappa) > 0$ ,  $r(\kappa) > 0$  for positive  $\kappa$ . We rewrite the complex frequency using new variables

$$\varsigma = \text{Re}(\varsigma) + i\text{Im}(\varsigma) = \sigma + i\omega. \quad (8)$$

Substituting (8) into general solution (4) gives

$$u \simeq u_0 e^{(\sigma + i\omega)t + i\kappa x} = u_0 e^{\sigma t} e^{i(\omega t + \kappa x)}, \quad (9)$$

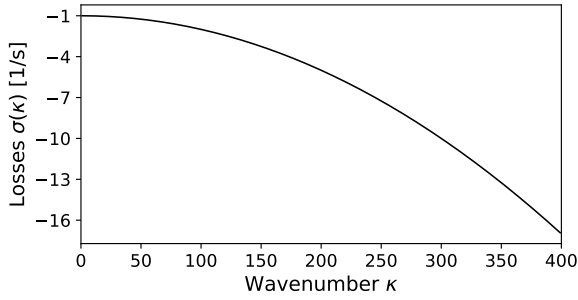


Figure 7. Losses  $\sigma(\kappa)$  corresponding to Eq. (3) shown for parameter values (12).

from here it is easy to see that imaginary part of roots (7) corresponds to oscillation frequencies, and real part

$$\sigma(\kappa) = -q = -\alpha - \beta\kappa^2, \quad (10)$$

to losses. Clearly, for real wavenumbers  $\kappa$  such that  $q^2 \leq r$  imaginary part

$$\omega(\kappa) = \sqrt{q^2 - r} = \sqrt{-(\alpha + \beta\kappa^2)^2 + c^2\kappa^2 + \gamma^2\kappa^4} \neq 0, \quad (11)$$

and the resulting string vibration corresponds to normal damped wave propagation. For realistic values of parameters in (3), the condition  $q^2 \leq r$  (traveling wave solution) holds for the vast majority of the audible frequency range. Also, notice that for  $\alpha, \beta \geq 0$ , loss  $\sigma = -q$  depends on  $\kappa$ , the damping rates are wavenumber and thus frequency dependent, moreover, the losses increase as a function of  $\kappa$ . On the other hand, if  $q^2 > r$ , then both roots (7) are purely real and non-positive, yielding damped non-traveling solutions. A more detailed analysis of this model has been performed by Bensa *et al.* in [6].

In order to demonstrate the obtained analytic results realistic parameter values, taken from [6], for Eq. (3) are chosen as follows:

$$c \simeq 200 \frac{\text{m}}{\text{s}}, \quad \gamma \simeq 1 \frac{\text{m}^2}{\text{s}}, \quad \alpha \simeq 1 \frac{1}{\text{s}}, \quad \beta \simeq 10^{-4} \frac{\text{m}^2}{\text{s}}. \quad (12)$$

These values correspond to a highly dispersive piano string rather than the guitar string considered below. In further discussion we are ignoring small wavenumber (extremely long wavelength) modes (in this case  $\kappa \lesssim 0.760$ )<sup>2</sup>. The behaviour for  $\kappa \lesssim 0.760$  is most likely non-physical due to the heuristics of the manner in which Eq. (3) was derived. Additionally, wave motion related to these wave components is outside the audible range of wavenumbers.

Figure 7 shows the decay curve for the selected parameters (12). As expected the exponential decay rates become greater as a function of  $\kappa$ . Phase velocity  $v_p(\kappa) = \omega/\kappa$  and group velocity  $v_g(\kappa) = d\omega/d\kappa$  curves are shown in Fig. 8. For all  $\kappa > 0$ ,  $v_g > v_p$  which means that with the passage of time a pulse propagating on the string will distort in a manner such that a high-frequency oscillating tail will tend to appear in front of the pulse.

<sup>2</sup> Criterion for determining the value: solve  $dv_g/d\kappa = 0$  for  $\kappa > 0$ .

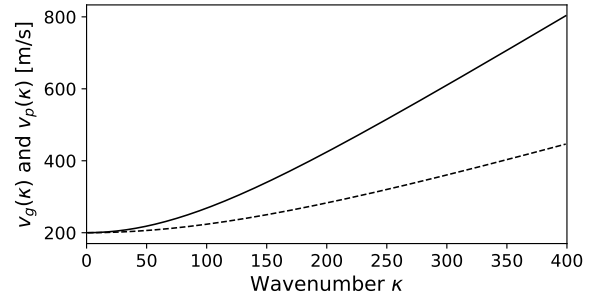


Figure 8. Group velocity, shown with the solid line, and phase velocity, shown with the dashed line, corresponding to Eq. (3) and calculated for parameter values (12).

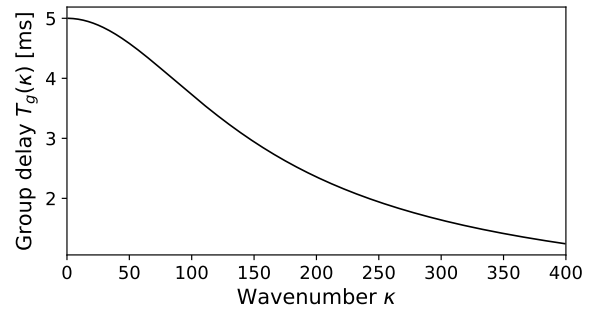


Figure 9. Group delay corresponding to Eq. (3) shown for parameter values (12).

This type of dispersion is referred to as the *anomalous* dispersion. The group delay, unit length multiplied by the inverse of the group velocity, shown in Fig. 9 also confirms that high-frequency wave components travel faster than the low-frequency ones. This behaviour can be confirmed by numerically integrating Eq. (3). The initial value problem is solved on an infinite half-plane  $x \in (-\infty, \infty)$ ,  $t \in [0, \infty)$  to eliminate any effects of wave interactions for  $t \gg 1$  caused by the fast traveling high-frequency wave components reflecting from the edges of a finite integration domain. We select a bell-shaped initial condition

$$u(x, 0) = A \operatorname{sech}^2 \eta x = \frac{4Ae^{2\eta x}}{(1 + e^{2\eta x})^2}, \quad (13)$$

where  $A = 2$  mm is the string amplitude,  $\eta = 2$  is the pulse width parameter. This parameter selection results in an approximately 2 m wide pulse — not too dissimilar from the realistic wavelengths found in string instruments. Figure 10 shows the integration result for parameter values (12) and for three space positions. The pulse evolution is exactly as predicted by the dispersion analysis. A dispersive high-frequency oscillating tail emerges in front of the main pulse by arriving earlier and becomes more prominent further the pulse propagates.

A careful look at the expressions of the phase and group velocities reveals that  $\lim_{\kappa \rightarrow \infty} v_p = \infty$  and  $\lim_{\kappa \rightarrow \infty} v_g = \infty$ . These results are clearly not physical. In a realistic physically-sound models these curves should plateau out to some finite *dynamic* velocity value. One could fix this *problem* by adding appropriate small magnitude higher order terms to model (3). Once again, as in the case  $\kappa \ll 1$ ,

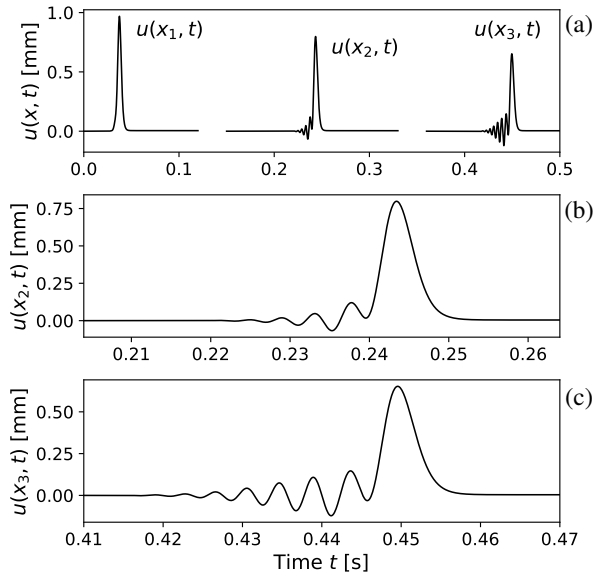


Figure 10. (a) Numerical integration of the initial value problem corresponding to Eq. (3), initial condition (13) and parameter values (12). Results are shown for space positions:  $x_1 = \pm 7.50$  m,  $x_2 = \pm 48.75$  m,  $x_3 = \pm 90.00$  m. (b) Magnified pulse shape, case  $x = x_2$ . (c) Magnified pulse shape, case  $x = x_3$ .

we conclude that this behaviour is non-physical and luckily for us outside the audible range of frequencies.

#### 4. NON-DISPERSIVE STRING MODEL

In order to identify the high-frequency and low-amplitude dispersive wave propagation in the experimental measurements presented below, we also consider a non-dispersive model with frequency-independent loss. The heuristics of our approach are directly determined by the d'Alembert formula (traveling wave solution). Modeling approach presented here is similar to [18, 19].

We consider vibration of a lossy ideal string described by wave equation

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} - 2\alpha \frac{\partial u}{\partial t}, \quad (14)$$

where  $u(x, t)$  is the displacement,  $c = \sqrt{T/\mu}$  is the speed of the waves traveling on the string,  $T$  is the tension and  $\mu$  is the linear mass density (mass per unit length) of the string. In the context of a real string Eq. (14) can be used as an approximation of thin homogeneous elastic string vibration under a small amplitude restriction. In this case wave speed  $c = \sqrt{T/(\rho A_o)}$ , where  $\rho$  is the volumetric density,  $A_o = \pi r^2$  is the cross-section area of a cylindrical string, and  $T$  is the tension. Second term on the right-hand side of (14) introduces frequency-independent loss, much the same way as in Eq. (3). It is easy to show that for  $\alpha > 0$  all frequency components will decay  $\sim e^{-\alpha t}$ . As in the case (3), this term can be seen as a perturbation term acting on the wave equation in the following form:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, \quad (15)$$

thus its linear effects on the final solution can be added separately. For now we focus on Eq. (15). It is well known that Eq. (15) has an analytical solution. For infinite string (ignoring boundary conditions for now), for initial conditions  $u(x, 0) = u_0(x)$ , and  $\partial u(x, 0)/\partial t = 0$  the solution takes the following form:

$$u(x, t) = \frac{1}{2} (u_0(x - ct) + u_0(x + ct)). \quad (16)$$

This solution represents a superposition of two traveling waves:  $u_0(x - ct)/2$  moving to the right (positive direction of the  $x$ -axis); and  $u_0(x + ct)/2$  moving to the left. Function  $u_0/2$  describes the shape of these waves and stays constant with respect to  $x$ -axis, as they are translated in opposite directions at speed  $c$ .

In general, a wave on any arbitrary segment of the string can be understood as a sum of two traveling waves that do not need to be equal. It can be written as

$$u(x, t) = r(x - ct) + l(x + ct), \quad (17)$$

where  $r(x - ct)$  is the traveling wave moving to the right and  $l(x + ct)$  is the traveling wave moving to the left.

A well-known time-stepping method for implementing d'Alembert formula is the following. We discretise  $xt$ -plane into  $n \times m$  discrete samples. We discretise the  $x$ -axis with grid spacing  $\Delta x = L/n$  where  $L$  is the speaking length of the string, and the  $t$ -axis with grid spacing  $\Delta t = t_{\max}/m$ , where  $t_{\max}$  is the integration time. We let  $x_i = i\Delta x$ , where  $0 \leq i \leq n$  and  $t^j = j\Delta t$ , where  $0 \leq j \leq m$ . From here it follows that  $u_i^j = u(x_i, t^j)$ ,  $r_i^j = r(x_i, t^j)$ , and  $l_i^j = l(x_i, t^j)$ . And, by applying

$$r_i^{j+1} = r_{i-1}^j, \quad (18)$$

$$l_i^{j+1} = l_{i+1}^j, \quad (19)$$

for all grid points  $i$  and  $j$  in a sorted order one gets translation of numerical values  $r_i^j$  and  $l_i^j$  propagating in opposite directions with respect to the  $x_i$ -axis. This result agrees with d'Alembert formula (16) or (17) and can be understood as a digital waveguide based on traveling wave decomposition and use of two delay lines. The equivalence between the model used here and digital waveguide modeling is shown in [20].

So far we have not addressed the boundary conditions of Eq. (15). We assume that the string is fixed at both ends. The following boundary conditions apply:

$$u(0, t) = u(L, t) = 0, \quad t \in [0, t_{\max}], \quad (20)$$

where  $t_{\max}$  is the desired integration time. By applying boundary conditions (20) to the general solution (17) The reflected traveling wave located at  $x = 0$  can be found in the following form:

$$u(0, t) = r(-ct) + l(ct) = 0 \Rightarrow r(-ct) = -l(ct), \quad (21)$$

and similarly for  $x = L$ :

$$\begin{aligned} u(L, t) = r(L - ct) + l(L + ct) = 0 \Rightarrow \\ \Rightarrow l(L + ct) = -r(L - ct). \end{aligned} \quad (22)$$

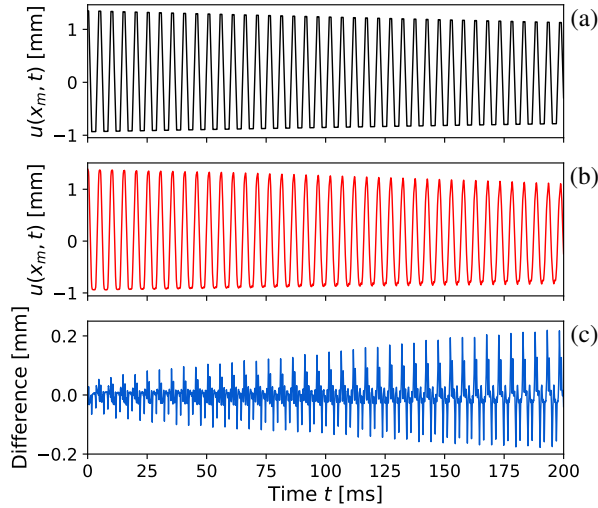


Figure 11. (a) Output of the model based on Eq. (14). (b) Experimental measurement. (c) Difference between model (14) and the experiment.

These results are discretised according to the discretisation scheme discussed above. The traveling wave (21) reflected from the left boundary at  $x = 0$  is

$$r_0^j = -l_0^j, \quad j \in [0, m], \quad (23)$$

and the traveling wave (22) reflected from the right boundary at  $x = L$  is

$$l_n^j = -r_n^j, \quad j \in [0, m]. \quad (24)$$

In order to obtain the resulting string displacement  $u_i^j$ , for the selected initial and boundary conditions, a superposition of traveling waves (18), (19), (23), and (24) is found in accordance with general solution (17)

$$u_i^j = r_i^j + l_i^j, \quad i \in [0, n], \quad j \in [0, m]. \quad (25)$$

Finally, there remains the question of loss introduced in (14). Since loss is  $\sim e^{-\alpha t}$  in the continuous domain and in the discrete domain  $\sim e^{-\alpha j \Delta t}$  we update (18) and (19) to

$$r_i^{j+1} = r_{i-1}^j e^{-\alpha j \Delta t / j} = r_{i-1}^j e^{-\alpha \Delta t}, \quad (26)$$

$$l_{i+1}^{j+1} = l_{i+1}^j e^{-\alpha j \Delta t / j} = l_{i+1}^j e^{-\alpha \Delta t}. \quad (27)$$

## 5. RESULTS AND DISCUSSION

Figure 5 shows the experimental set-up schematically. The following values of parameters were used: speaking length of the string  $L = 0.65$  m; fundamental frequency  $f_0 = 196.36$  Hz; excitation point for triangular initial condition  $x = x_e = 0.25L = 0.163$  m; initial amplitude  $A = 1.76$  mm; loss parameter  $\alpha = 1.1 \text{ s}^{-1}$ . All time and frequency domain results are shown or calculated for string displacement  $u(x_m, t)$  where measurement point  $x_m = 0.41L = 0.266$  m. The spectrograms and power spectra are calculated using the Fast Fourier Transform algorithm. In calculating spectrograms a sliding window approach, in combination with the Hanning window function are used. Here,

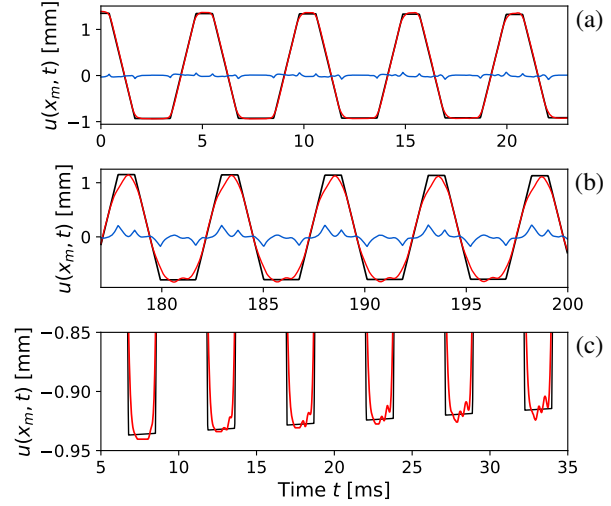


Figure 12. Magnified time-series of the results shown in Fig. 11. Model (14) output shown with the black line, the experiment shown with the red and the difference with the blue line. (a) First 25 ms of vibration. (b) Last 25 ms of vibration. (c) Magnification of the convex valleys of the signals shown in Fig. 11 displayed for  $5 \leq t \leq 35$  ms.

window size is 70 ms and window overlap value is 20% of the window size. The Short Time Fourier Transform (STFT) spectrogram is calculated using window size 1.4 ms and the overlap value is 95%.

Figures 11 and 12 show the time domain results. A comparison of the simulated vibration, based on model (14), to the experiment is shown for the first 200 ms of vibration. The presented waveforms match up relatively well, given the simplicity of the model (14), especially at the beginning of the vibration. The differences between the presented results (blue lines) are growing with the passage of time which means that all processes not described by model (14) are progressively accumulating. We remind that the dispersion is a progressively accumulating phenomenon. Naturally, we consider two candidates for these unidentified processes: the anomalous dispersion, and the frequency-dependent loss as described by the more realistic full model (3).

Let us consider the possibility of dispersion. We assume that the experimental data has losses similar to (10) of full model (3). The losses associated with the large wavenumbers (see Fig. 7), remain non-dominating for the first periods of vibration, in fact, that is clearly evident in Fig. 12a where the model (14) is almost equal to the experiment. Small differences are present only for the discontinuous edges of the peaks and valleys of the modeled time-series. Not surprisingly, these regions are associated with extremely large wavenumbers  $\kappa$ . As long as we focus on the regions in-between the pointy edges, the losses should be minimal, especially for  $t \ll 1$ . Figure 12c shows the experimental evidence of the anomalous dispersion. The evolution of the vibration is qualitatively similar to the result shown in Fig. 10. A dispersive high-frequency oscillating tail emerges from the right-hand side of the signal's valley and propagates to the left. This happens for every succeed-



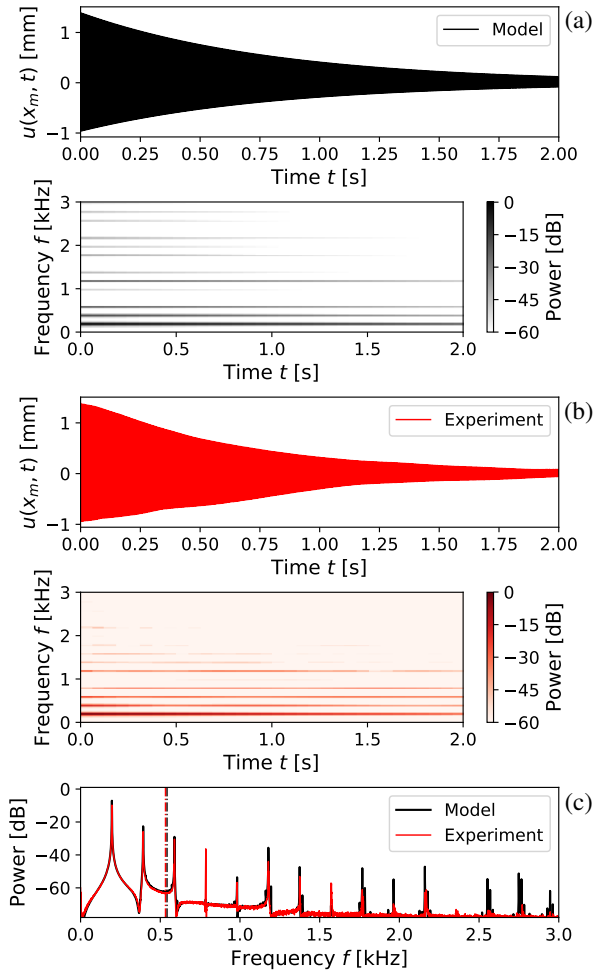


Figure 13. (a) Modeled time-series (14) and the corresponding spectrogram. (b) Time-series and spectrogram of the experiment. (c) Power spectra of the above signals. Spectral centroids are indicated by the dash-dotted lines.

ing period with the oscillation amplitude becoming progressively larger. Although, the geometry (boundary conditions) of the problem discussed here compared to the one shown in Fig. 10 is different the conclusions regarding the leading high-frequency tail evolution still hold.

Figure 13 shows the frequency domain results. At the beginning of the vibration partial contents of the modeled signal (14), in comparison to the experiment, has higher peaks for  $f \gtrsim 1$  kHz. This is most likely due to a combination of the absence of the frequency-dependent attenuation in model (14) and the unrealistic discontinuities present in the initial condition shown in Fig. 5. The decay rate of high-frequency partials, as seen on the spectrograms, is greater for the experimental result which agrees with the dispersion analysis of full model (3). No obvious high-power inharmonic partials are visible in the power spectra shown in Fig. 13c. The identified dispersive wave features present in the experimental data are extremely weak due to the low bending stiffness of the thin guitar string used in the experiment. Additionally, the dispersion is also masked by the frequency-dependent losses. The attenu-

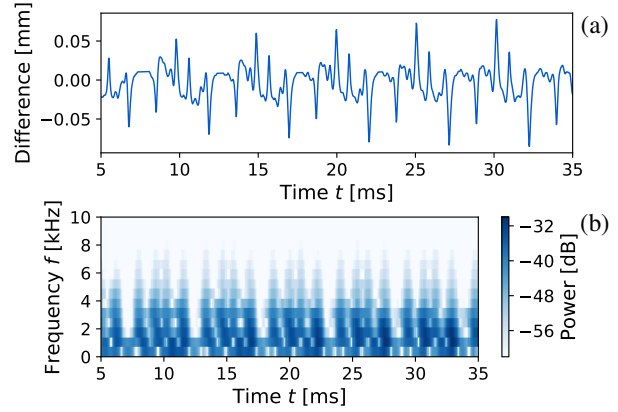


Figure 14. (a) Magnified time-series of the difference between the model (14) and the experiment shown in Fig. 11c. (b) STFT spectrogram of the above signal.

ation is particularly overwhelming for large wavenumber modes, associated with our dispersive oscillating leading tail. This means that with the passage of time, the high-frequency oscillations simply decay much faster compared to the low-frequency modes.

Figure 14 shows another line of evidence for the existence of dispersion. It presents the STFT spectroscopic analysis of the difference signal shown in Fig. 11c. This way of visualising effects of dispersion was suggested by Woodhouse [21]. The window size of 1.4 ms is related to the highest partial present in the signal ( $\approx 7$  kHz). The distinct vertical “formants” can be seen slanting to the left. This result in combination with the time domain result shown in Fig. 12c indicates that high-frequency wave components arrive sooner in comparison to the low-frequency ones.

It is natural to treat the numerical approach presented in Sec. 4 as a digital waveguide and apply digital filters to the traveling waves (18) and (19). All-pass dispersive filters could be used to tune our time-stepping model to the experimental data and thus synthesise realistic sounds [22–24]. If one wishes to remain true to the full model (3) it is possible to derive a digital filter based on it. Bensa *et al.* in [6] show how one can relate the full model to a digital waveguide structure using dispersion relations (10) and (11).

## 6. CONCLUSIONS

This paper presented results of the experimental study of dispersive wave propagation on guitar strings. The evidence of the dispersion was found and presented. Unsurprisingly, the effect was minor but definitively present.

The high-resolution experimental data of the string displacement was gathered using the video-kymographic high-speed imaging. The experimental data was then compared against the non-dispersive model described by Eq. (14) that was used to identify dispersive features present in experimental data, and to distinguish them from other effects, such as frequency-dependent dissipation shown to be prominent in the more realistic model described by Eq. (3).

The video-kymographic experimental method presented here has proven to be highly reliable for our purposes. We

strongly suggest to use this method for measurements of rapidly moving sub-millimetre sized object and displacements in applications where high spatial and temporal resolution of measurement results are required.

### Acknowledgments

The first two authors are supported by the Estonian Ministry of Edu. and Res. (IUT33-24) and the Doctoral Studies and Int. Prog. DoRa Plus, Action 1 (Archimedes Foundation, Estonia). This research is related to the Nordic Sound and Music Computing Network — NordicSMC (NordForsk project no. 86892). The authors are grateful to the Aalto Uni. funding scheme for infrastructure. The measurements for this study were made in Dec. 2018, when D. Kartofelev and J. Arro visited the Aalto Acoustics Lab.

### 7. REFERENCES

- [1] M. Podlesak and A. R. Lee, "Dispersion of waves in piano strings," *J. Acoust. Soc. Am.*, vol. 83, no. 1, 1988.
- [2] L. Hiller and P. M. Ruiz, "Synthesizing musical sounds by solving the wave equation for vibrating objects," *J. Audio Eng. Soc.*, vol. 19, no. 6, pp. 462–551, 1971.
- [3] R. Bacon and J. Bowsheer, "A discrete model of a struck string," *Acustica*, vol. 41, no. 1, pp. 21–27, 1978.
- [4] A. Chaigne, "On the use of finite differences for musical synthesis. Application to plucked stringed instruments," *J. Acoust.*, vol. 5, pp. 181–211, 1992.
- [5] A. Chaigne and A. Askenfelt, "Numerical simulations of struck strings. I. A physical model for a struck string using finite difference methods," *J. Acoust. Soc. Am.*, vol. 95, no. 2, pp. 1112–1118, 1994.
- [6] J. Bensa, S. Bilbao, R. Kronland-Martinet, and J. O. Smith, "The simulation of piano string vibration: from physical models to finite difference schemes and digital waveguides," *J. Acoust. Soc. Am.*, vol. 114, no. 2, pp. 1095–1107, 2003.
- [7] C. Cuesta and C. Valette, "Evolution temporelle de la vibration des cordes de clavecin," *Acustica*, vol. 66, no. 1, pp. 37–45, 1988.
- [8] C. Desvages, S. Bilbao, and M. Ducceschi, "Improved frequency-dependent damping for time domain modelling of linear string vibration," in *Proc. 22nd Int. Cong. on Acoust.*, Buenos Aires, Argentina, Sept. 2016, pp. 1–10.
- [9] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*. USA: Springer, 1998.
- [10] R. C. D. Paiva, J. Pakarinen, and V. Välimäki, "Acoustics and modeling of pickups," *J. Audio Eng. Soc.*, vol. 60, no. 10, pp. 768–782, 2012.
- [11] J. Pakarinen and M. Karjalainen, "An apparatus for measuring string vibration using electric field sensing," in *Proc. Stockholm Music Acoust. Conf.*, Stockholm, Sweden, 2003, pp. 739–742.
- [12] J. Kotus, P. Szczuko, M. Szczodrak, and A. Czyżewski, "Application of fast cameras to string vibrations recording," in *2015 Signal Processing: Algorithms, Architectures, Arrangements, and Applications*, Poznan, Poland, September 2015, pp. 104–109.
- [13] Y. Achkire and A. Preumont, "Optical measurement of cable and string vibration," *Shock and Vibration*, vol. 5, no. 3, pp. 171–179, 1998.
- [14] M. Podlesak and A. R. Lee, "A photovoltaic detector for string vibration measurement," *J. Acoust. Soc. Am.*, vol. 79, no. 6, pp. 2092–2093, 1986.
- [15] D. Kartofelev, M. Mustonen, A. Stulov, and V. Välimäki, "Application of high-speed line scan camera for string vibration measurements," in *Proc. Int. Symp. on Musical Acoust.*, Le Mans, France, July 2014, pp. 629–634.
- [16] M. Mustonen, D. Kartofelev, A. Stulov, and V. Välimäki, "Application of high-speed line scan camera for acoustic measurements of vibrating objects," in *Proc. 7th Forum Acusticum*, Kraków, Poland, September 2014, pp. 1–6.
- [17] M. Mustonen, D. Kartofelev, A. Stulov, and V. Välimäki, "Experimental verification of pickup non-linearity," in *Proc. Int. Symp. on Musical Acoust.*, Le Mans, France, July 2014, pp. 651–656.
- [18] D. Kartofelev, A. Stulov, H.-M. Lehtonen, and V. Välimäki, "Modeling a vibrating string terminated against a bridge with arbitrary geometry," in *Proc. 4th Stockholm Music Acoust. Conf.*, Stockholm, Sweden, Jul. 30–Aug. 3 2013, pp. 626–632.
- [19] D. Kartofelev, "Kinematics of ideal string vibration against a rigid obstacle," in *Proc. 20th Int. Conf. on Digital Audio Effects*, Edinburgh, Scotland, UK, September 2017, pp. 40–47.
- [20] J. O. Smith, *Physical Audio Signal Processing for Virtual Musical Instruments and Audio Effects*. W3K Publishing, 2010.
- [21] J. Woodhouse, "Plucked guitar transients: comparison of measurements and synthesis," *Acta Acust. united Ac.*, vol. 90, no. 5, pp. 945–965, 2004.
- [22] J. S. Abel and J. O. Smith, "Robust design of very high-order allpass dispersion filters," in *Proc. Int. Conf. on Digital Audio Effects*, Montreal, Canada, September 2006, pp. 13–18.
- [23] J. Rauhala and V. Välimäki, "Tunable dispersion filter design for piano synthesis," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 253–256, 2006.
- [24] —, "Dispersion modeling in waveguide piano synthesis using tunable allpass filters," in *Proc. Int. Conf. on Digital Audio Effects*, Montreal, Canada, September 2006, pp. 71–76.

# Real-Time Modeling of Audio Distortion Circuits with Deep Learning

Eero-Pekka Damskäg, Lauri Juvela, and Vesa Välimäki

Acoustics Lab, Department of Signal Processing and Acoustics

Aalto University, Espoo, Finland

eero-pekka.damskagg@aalto.fi

## ABSTRACT

This paper studies deep neural networks for modeling of audio distortion circuits. The selected approach is black-box modeling, which estimates model parameters based on the measured input and output signals of the device. Three common audio distortion pedals having a different circuit configuration and their own distinctive sonic character have been chosen for this study: the Ibanez Tube Screamer, the Boss DS-1, and the Electro-Harmonix Big Muff Pi. A feedforward deep neural network, which is a variant of the WaveNet architecture, is proposed for modeling these devices. The size of the receptive field of the neural network is selected based on the measured impulse-response length of the circuits. A real-time implementation of the deep neural network is presented, and it is shown that the trained models can be run in real time on a modern desktop computer. Furthermore, it is shown that three minutes of audio is a sufficient amount of data for training the models. The deep neural network studied in this work is useful for real-time virtual analog modeling of nonlinear audio circuits.

## 1. INTRODUCTION

Guitar distortion effects are traditionally based on analog audio circuitry. These circuits contain nonlinear components, such as diodes, transistors or triodes to produce the desired distortion effect. As most of music production today is carried out using digital audio workstations (DAWs), there is an increasing demand for faithful digital emulations of analog audio effects. The field of virtual analog (VA) modeling is concerned with creating these digital emulations, which allow musicians to record and produce music without investing in expensive analog equipment.

A common approach for VA modeling of distortion effects is “white-box” modeling [1–4]. White-box modeling is based on analysis and discrete-time simulation of the analog circuitry. If the circuit and the characteristics of its nonlinear components are known, white-box modeling can be very accurate. However, circuit simulation can get computationally demanding when there are many reactive

components and nonlinear elements in the circuit, and the involved design process can be labor intensive.

An alternative approach for VA modeling is “black-box” modeling. Black-box modeling is based on measuring the circuit’s response to some input signals, and creating a model which replicates the observed input-output mapping. Black-box models for VA modeling include block-oriented models, which are based on assumptions about the design of the modeled circuit [5–9]. As an example, a Wiener model [5, 8] emulates the circuit as a linear filter followed by a static nonlinearity. Other black-box modeling methods include Volterra series models [10, 11], dynamical convolution [12] and kernel regression [13].

In our previous work, a deep neural network for black-box modeling of nonlinear audio circuits was presented, and applied to the modeling of a vacuum tube amplifier [14]. The model is based on the WaveNet convolutional neural network [15]. The proposed neural network model is made up of a series of convolutional layers, which consist of a filter followed by a nonlinear activation function. As the filtering and nonlinear processing are applied in several stages, the neural network should be suitable for modeling of a broad range of nonlinear audio circuits.

This work follows the previous work with an emphasis on the real-time performance of the model. Three guitar distortion pedals are modeled in this work: the Ibanez Tube Screamer, the Boss DS-1, and the Electro-Harmonix Big Muff Pi. A hyperparameter search is conducted to find a suitable trade-off between modeling accuracy and computational load. Experiments are carried out to find the minimum amount of data required for successful training. Finally, a low-latency implementation of the proposed deep neural network, which can be run in real time on a consumer-grade computer, is presented.

The rest of this paper is structured as follows. Section 2 provides background on the modeled distortion effects. Section 3 details the proposed deep neural network for black-box modeling. In Section 4, the developed real-time implementation of the model is presented. In Section 5, the hyperparameter search and its results are detailed, and the effect of the amount of training data on the modeling accuracy is examined. Section 6 presents the modeling results. Finally, Section 7 concludes the paper.

## 2. MODELED DEVICES

Three guitar distortion effects are considered in this study: the Ibanez Tube Screamer, the Boss DS-1, and the Electro-Harmonix Big Muff Pi. Detailed circuit analyzes of all

Copyright: © 2019 Eero-Pekka Damskäg et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

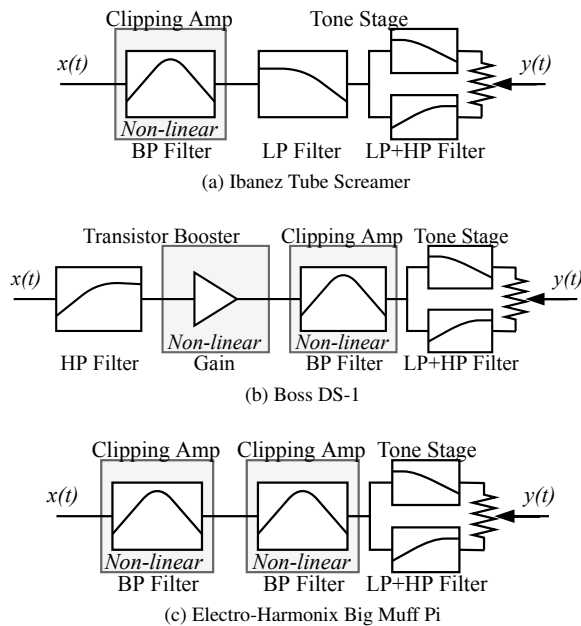


Figure 1: Block diagrams of the distortion effects.

three pedals can be found online [16].

### 2.1 Ibanez Tube Screamer

The Ibanez Tube Screamer is one of the most well known guitar overdrive pedals. There have been several reissues of the pedal since the release of the original TS808 in the late 1970s [17]. For this study, the TS7 version, which was introduced in the early 2000s, was used. Digital models for the Tube Screamer have been previously proposed by Yeh *et al.* [1, 18], Werner *et al.* [4], and Eichas *et al.* [8].

The simplified structure of the Tube Screamer pedal is shown in Figure 1a. The nonlinear behavior of the pedal occurs in the clipping amp. It is an op-amp-based bandpass filter with diodes in the feedback path of the op amp. After the clipping amp, there is the tone stage, which consists of a passive lowpass filter followed by an active filter, which can act as a low-pass or a high-pass filter depending on the position of the tone potentiometer.

### 2.2 Boss DS-1

The Boss DS-1 is a famous distortion pedal released in the late 1970s [16]. Its nonlinear characteristics resemble those of a hard clipper. Before this work, digital models for the DS-1 have been proposed by Yeh *et al.* [2, 18].

The DS-1 has two nonlinear stages, as shown in Figure 1b. The transistor booster stage performs high-pass filtering and amplification of the input signal. Nonlinearities are introduced to the signal when the boosted peak-to-peak voltage of the signal exceeds the 9V supply voltage.

The actual distortion effect is produced by the clipping amp. The clipping amp is an op-amp-based bandpass filter with two diodes shunting the output signal to ground. This placing of the diodes introduces a hard-clipping effect. This is in contrast to the soft-clipping effect produced by placing the diodes in the feedback path, as in the Tube

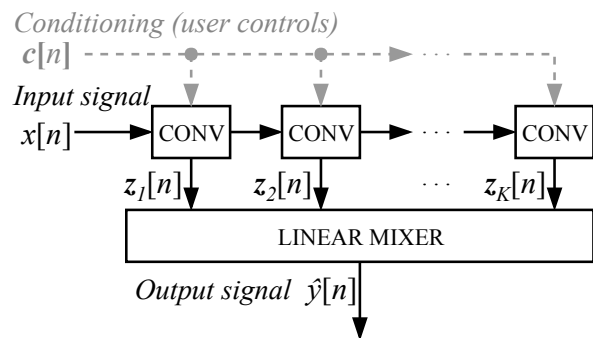


Figure 2: Proposed deep neural network model.

Screamer [16]. The tone stage has passive low-pass and high-pass filters whose outputs are mixed based on the setting of the tone knob. Setting the tone knob to the middle position results in a bandstop response, with a center frequency at approximately 500 Hz [16].

### 2.3 Electro-Harmonix Big Muff Pi

The Big Muff Pi is a distortion/fuzz effect known for its distinctive long-sustain sound [16, 19]. Electro-Harmonix began mass-producing the pedal in the early 1970s. Since then, various models have been released with different exteriors and with slight circuit modifications [19]. Digital models for the Big Muff have been proposed [8, 20]

A simplified block diagram of the Big Muff is shown in Figure 1c. The circuit has two identical clipping amps in series. However, in some versions, the two clipping amp circuits have different component values, such as different collector resistors [19]. The clipping amp is a transistor-based bandpass filter. As with the Tube Screamer, the clipping in the Big Muff is produced by two diodes placed in the feedback path. The combined effect of the two cascaded soft-clipping amps is hard clipping. The tone stage is similar to the one in the Boss DS-1.

## 3. DEEP NEURAL NETWORK MODEL

The proposed model for black-box modeling is based on the WaveNet neural network [15]. The original WaveNet is a convolutional autoregressive model, where the previous output sample is fed back to the model for making the next prediction. In our previous work, a feedforward variant of the WaveNet architecture was presented and applied to modeling of a vacuum tube amplifier [14].

The proposed model is shown in Figure 2. The neural network consists of a series of convolutional layers. The raw input waveform is given as input to the first convolutional layer. The convolutional layers apply linear filtering and a nonlinear activation function to the signal.

Optionally, the output of the network can be conditioned on user controls. In the previous work [14], the gain setting of the vacuum tube amplifier was fed to the model along with the input signal, allowing the model to represent different playing configurations of the amplifier. In the experiments of this work, the conditioning is left out,



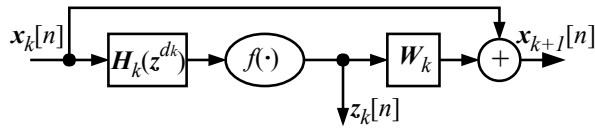


Figure 3: Block diagram of a single convolutional layer.

since we are measuring physical devices, and automatic knob adjustment and data collection is left for future work.

In the previous work, the outputs of the convolutional layers were fed to a three layer “post-processing module” with  $1 \times 1$  convolutions and nonlinear activation functions. In convolutional neural network terminology, a  $1 \times 1$  convolution refers to a matrix multiplication applied at each time step in the signal. In this work, the post-processing module is replaced by a linear mixer, i.e., a single linear  $1 \times 1$  convolution layer. According to our experiments, the network performs similarly or better with the linear output layer, while reducing the complexity and the computational load of the network.

### 3.1 Convolutional Layer

The convolutional layer used in the model is shown in Figure 3. The input signal is first processed by the dilated causal FIR filter  $H_k(z^{d_k})$ , where  $k$  is the layer index and  $d_k$  is the integer-valued “dilation factor” of the filter. Since the convolutional layers generally have multiple channels, the filtering is performed as a multiple-input and multiple-output (MIMO) convolution with a kernel  $\mathbf{H}_k$ . This means that a filter is learned for each pair of input and output channels. The individual filters in the kernel have impulse responses

$$h[n] = \sum_{m=0}^{M-1} w_m \delta[n - md_k], \quad (1)$$

where  $\delta[n]$  is the Kronecker delta function, and  $w_m$  are the non-zero coefficients of the filters learned by the network.

Next, a nonlinear activation function  $f(\cdot)$  is applied to the biased convolution output, producing the layer output

$$z_k[n] = f[(\mathbf{H}_k * \mathbf{x}_k)[n] + \mathbf{b}_k], \quad (2)$$

where  $*$  denotes the convolution operator, and  $\mathbf{b}_k$  is the learned bias term.

The layers include a residual connection, which means that the input to the next layer is

$$\mathbf{x}_{k+1}[n] = \mathbf{W}_k z_k[n] + \mathbf{x}_k[n], \quad (3)$$

where the  $1 \times 1$  convolution kernel  $\mathbf{W}_k$  controls the mixing between the layer input  $\mathbf{x}_k$  and the layer output  $z_k$  before the next layer.

Each convolutional layer is a Wiener model: a linear filter followed by a static nonlinearity. Conventional black-box approaches are often based on a Wiener [5, 8], a Hammerstein [6] or a Wiener-Hammerstein [7, 9] model assumption. As a cascade of Wiener models, the proposed neural network makes fewer assumptions about the design of the modeled device, and is expected to be applicable to

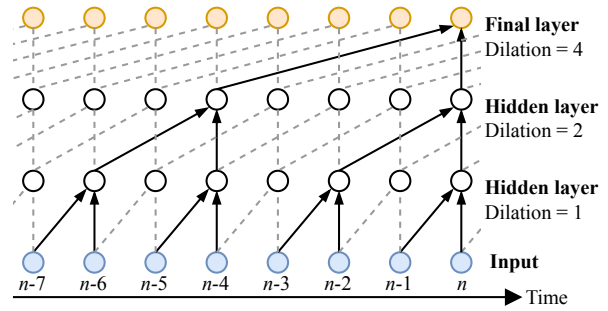


Figure 4: Visualization of three convolutional layers in series and the resulting receptive field of  $N = 8$ . The figure has been adapted from [15].

the modeling of a broad range of nonlinear systems. Furthermore, the deep learning approach optimizes the system response jointly, and not block-by-block, so not only the model but also the optimization makes fewer assumptions about the behavior of the device under study.

### 3.2 Receptive Field

The proposed neural network is modeling the device under study in a feedforward fashion. The predicted output sample at a time instant  $n$  depends only on the  $N$  latest input samples, where  $N$  is called the receptive field of the model, or the order of the feedforward model. The receptive field depends on the number of convolutional layers, and the lengths of the filters in the layers. This is illustrated in Figure 4. The example network has 3 convolutional layers with dilation factors  $d_k = \{1, 2, 4\}$ , and  $M = 2$  non-zero coefficients for each filter. It can be seen that in this case, the current output sample depends on eight latest input samples. That is, the network has a receptive field of  $N = 8$ . Generally, the receptive field is given by

$$N = (M - 1) \sum_{k=1}^K d_k + 1, \quad (4)$$

where  $K$  is the number of convolutional layers. By increasing the dilation by a factor of two in each layer, the model order can be increased to thousands of samples with relatively few layers, allowing feedforward modeling of systems with long impulse responses.

To estimate the required receptive field for modeling of the distortion effects, their linear impulse responses were estimated using the swept-sine technique [6, 21]. A low-level sine sweep was used in order to minimize the effect of circuit nonlinearities in the measurement. The estimated lengths of the impulse responses were approximately 35 ms for the Big Muff, and approximately 45 ms for the Tube Screamer and the DS-1. With a 44.1-kHz sample rate, these correspond to required receptive fields of approximately 1500 to 2000 samples, respectively.

### 3.3 Loss Function

The neural network was trained by minimizing the error-to-signal ratio (ESR) with respect to the training data. Dur-

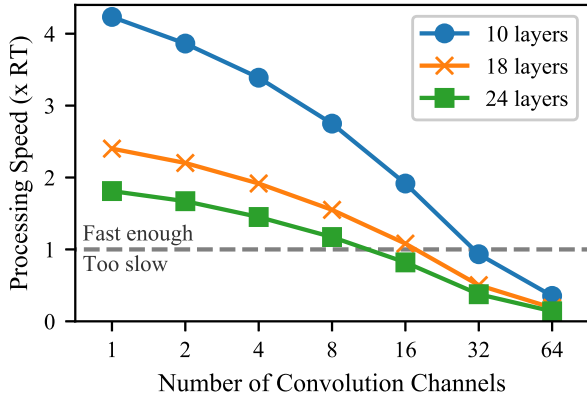


Figure 5: The processing speeds of models with different numbers of layers and convolution channels. The models use the gated activation. The cases above the horizontal dashed line can run in real time.

ing training and validation, a “pre-emphasis” filter was applied to the output and target signals before computing the ESR. For the  $i$ th training example, the pre-emphasized ESR is given by

$$\mathcal{E}^{\{i\}} = \frac{\sum_{n=-\infty}^{\infty} |y_p^{\{i\}}[n] - \hat{y}_p^{\{i\}}[n]|^2}{\sum_{n=-\infty}^{\infty} |y_p^{\{i\}}[n]|^2}, \quad (5)$$

where  $y_p^{\{i\}}$  is the pre-emphasized target signal, and  $\hat{y}_p^{\{i\}}$  is the pre-emphasized neural network output. The ESR can be considered as an energy-normalized sum-of-squares error. Without the energy normalization, the segments in the training data with most energy would dominate the loss.

The pre-emphasis filter was chosen as the first-order high-pass filter with transfer function

$$H(z) = 1 - 0.95z^{-1}, \quad (6)$$

which is very commonly used in speech processing [22]. The purpose of the filtering is to emphasize middle and high frequencies in the loss function. According to our experiments, the neural network struggles at modeling the high-frequency content introduced by the distortion effects without the pre-emphasis filtering.

#### 4. REAL-TIME IMPLEMENTATION

The proposed black-box models were implemented in C++, because the goal was to run the optimized model in real time. The real-time application was built using the open source JUCE framework. JUCE allows building cross-platform audio applications as well as VST, AU, and AAX plugins from a single source code. The Eigen library was used for matrix and vector operations. The source code is available at <https://github.com/damskagep/WaveNetVA>.

The C++ implementation of the deep neural network does not currently support model training. Instead, the models are trained using the Tensorflow library. The model hyperparameters and the values of the learned convolution kernels and biases are stored to a JSON file. The trained models can then be loaded to the C++ application.

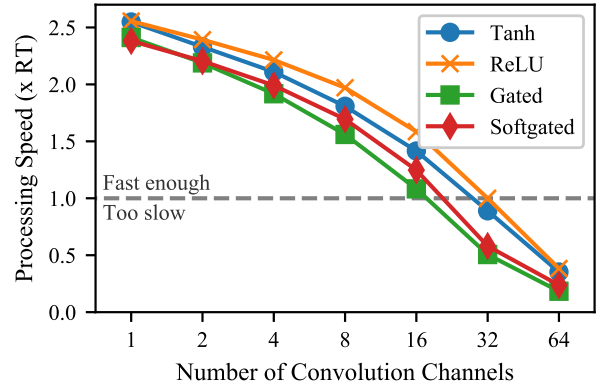


Figure 6: The processing speeds of the 18-layer model with different activation functions and different numbers of convolution channels. The cases above the horizontal dashed line can run in real time.

The real-time performance of the C++ code was estimated for several model configurations. The models were tested using an Apple iMac with an 2.8 GHz Intel Core i5 processor, using a short processing buffer of 64 samples and a sample rate of 44.1 kHz. During the test, all other applications were shut down and the computer was disconnected from the internet. This was done to minimize the effect of other processes in the test.

Figure 5 shows the processing speed of the model with different numbers of layers and different numbers of convolution channels. The models use the gated activation. The processing speed is expressed as a factor of the requirement for real-time application. Clearly, the computational load increases as the number of layers and channels is increased. The largest model running in real time uses 18 layers and 16 channels in the convolutional layers. With 24 layers, a model with 8 convolutional channels can be run in real time.

Figure 6 shows the processing speed of the 18-layer model using different activation functions. The activation functions are detailed in Section 5.2.2. The rectified linear unit (ReLU) is the computationally cheapest and the gated activation is the most computationally expensive activation.

#### 5. EXPERIMENTS

For the experiments described in the following, the neural networks were trained using the Adam optimizer [23]. The validation error was computed after each epoch. Early stopping was used with a patience of 20 epochs. The training data was split into 100 ms training examples, and a mini-batch size of 40 was used. A sample rate of 44.1 kHz was used in the experiments.

##### 5.1 Dataset Generation

Training data was generated by processing audio through the modeled distortion effects. The devices were measured using an audio interface connected to a computer via USB. One output of the audio interface was connected to the input of the measured device. The output of the device was

recorded by connecting it to one of the inputs of the audio interface. The output of the audio interface was also directly connected to another input of the audio interface, in order to estimate the effect of the audio interface in the measurement, as suggested in [9]. The recorded direct signal from the audio interface and the recorded output from the device under study make up the input/target pairs used in the training of the network.

The input sounds processed through the device were obtained from the guitar and bass guitar datasets<sup>1 2</sup> described in [24, 25], respectively. A random subset with 5 minutes of audio was picked from the datasets, with 2.5 minutes of guitar and 2.5 minutes of bass sounds. The data generated using these sounds was used for training. An additional minute of audio was randomly selected for validation. For testing of the networks, the test set signals from the previous work were reused [14].

All three modeled devices have a knob to control the intensity of the distortion effect and a “Tone” knob to control the filter in the tone stage. For the measurements, all knobs were set to the 12 o’clock, or middle, position. Filtering occurs in the tone stages of all pedals even when the Tone knob is set to the middle position [16]. That is, the middle position of the knob does not indicate an allpass setting for the filters in the tone stages.

## 5.2 Model Selection

The performance of the proposed neural network depends mostly on the number of channels used in the convolutional layers, the activation function, and the dilation pattern. The choice of these hyperparameters also affects the computational load of the model, as shown in Section 4. Therefore, a hyperparameter search was conducted to find a suitable trade-off between model performance and computational load.

### 5.2.1 Dilation Pattern

Three different dilation patterns were considered:

$$\begin{aligned} d_k &= \{1, 2, 4, \dots, 512\}, \\ d_k &= \{1, 2, 4, \dots, 256, 1, \dots, 256\}, \text{ and} \\ d_k &= \{1, 2, 4, \dots, 128, 1, \dots, 128, 1, \dots, 128\}. \end{aligned}$$

These dilation patterns correspond to models with 10, 18, and 24 convolutional layers, respectively. The number of non-zero coefficients in the filters was set to  $M = 3$ , which means that, according to Eq. (4), the 10, 18, and 24-layer networks have the receptive fields of  $N = 2047, 2045$ , and 1530 samples, respectively. At the 44.1-kHz sample rate, these receptive fields correspond to approximately 46, 46, and 35 ms, respectively.

### 5.2.2 Activation Functions

For the convolutional layers, the performance of the following activation functions are compared: the hyperbolic

tangent:

$$z = \tanh(\mathbf{H} * \mathbf{x}), \quad (7)$$

the rectified linear unit (ReLU):

$$z = \max(0, \mathbf{H} * \mathbf{x}), \quad (8)$$

and the gated activation, which was used in the original WaveNet [15]:

$$z = \tanh(\mathbf{H}_f * \mathbf{x}) \odot \sigma(\mathbf{H}_g * \mathbf{x}), \quad (9)$$

where  $\odot$  is the element-wise multiplication operation,  $\sigma(\cdot)$  is the logistic sigmoid function,  $\mathbf{H}_f$  and  $\mathbf{H}_g$  are the filter and gate convolution kernels, respectively. Finally, the softsign-gated activation, as used in [26], was evaluated:

$$z = g(\mathbf{H}_f * \mathbf{x}) \odot g(\mathbf{H}_g * \mathbf{x}), \quad (10)$$

where the hyperbolic tangent and the logistic sigmoid of the standard gated activation are both replaced by the softsign function:

$$g(x) = \frac{x}{1 + |x|}. \quad (11)$$

The softsign nonlinearity can be computationally cheaper than the hyperbolic tangent and the logistic sigmoid functions, as shown in Section 4, while having a similar shape.

With the gated activations, the convolutional layer used in the model can no longer be considered a Wiener model. Instead, it can be described as two parallel Wiener models, whose outputs are multiplied together to produce the layer output.

### 5.2.3 Results

In the following, the results of the hyperparameter search are presented. As there is an interest on the real-time performance of the models, the validation loss is shown as a function of the processing speed on the developed real-time C++ implementation of the model.

The effect of the choice of dilation pattern on the validation loss is shown in Figure 7. The validation loss is reported as an average loss over all the modeled devices. All models shown in Figure 7 use the gated activation, as given by Eq. (9). The number of convolution channels was varied with values 2, 4, 8, 16, and 32. It can be seen that the 10-layer model performs favorably with respect to the processing speed, while still obtaining a relatively low ESR. The 10-layer model with 16 channels has an average ESR of 4.2%, and runs 1.9 times faster than real time. The 18-layer model with 16 convolution channels had the lowest ESR of the models which run faster than real time. The model has an average ESR of 3.1%, and runs 1.1 times faster than real time.

Overall, the 24-layer model performs more poorly than the 18-layer model. It is possible that this is because the receptive field of the 24-layer model is slightly shorter than the estimated impulse response lengths of the Ibanez Tube Screamer and the Boss DS-1.

The effect of the choice of activation function is shown in Figure 8. The models shown in Figure 8 use 18 layers and the number of convolution channels was again varied

<sup>1</sup> [https://www.idmt.fraunhofer.de/en/business\\_units/m2d/smt/guitar.html](https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/guitar.html)

<sup>2</sup> [https://www.idmt.fraunhofer.de/en/business\\_units/m2d/smt/bass\\_lines.html](https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/bass_lines.html)

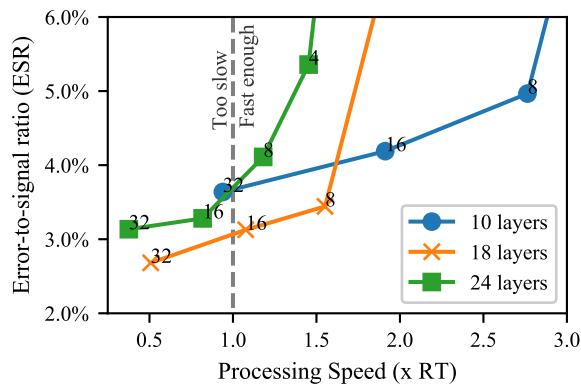


Figure 7: The validation error-to-signal ratio (ESR) as a function of the processing speed, using different numbers of layers and convolution channels. All models shown use the gated activation. The number of convolution channels used is indicated next to each model.

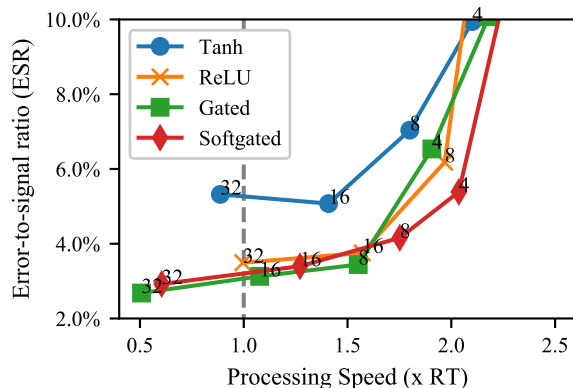


Figure 8: The validation error-to-signal ratio (ESR) as a function of processing speed with different activation functions and different numbers of channels in the convolutional layers. The number of convolution channels used is indicated next to each model.

with values 2, 4, 8, 16, and 32. It can be seen that the hyperbolic tangent activation performs worst out of all activations. The other activation functions perform similarly with each other. This suggests that the ReLU, the softsign-based gated activation and the standard gated activation are all viable options for a real-time application.

Based on the hyperparameter search, three models were chosen for the final evaluation. The hyperparameters of the selected models are shown in Table 1. Only models which run faster than real time were selected. WaveNet1 is the fastest of the selected models, and it has the worst ESR on the validation data. WaveNet3 is the slowest model, and it has the best ESR on the validation data. WaveNet2 is an intermediate model.

### 5.3 Training Data Length

An interesting question regarding neural networks for virtual analog modeling is the amount of data required for

Table 1: Hyperparameters of selected neural networks.

Model	WaveNet1	WaveNet2	WaveNet3
Activation	Gated	Gated	Gated
Layers	10	18	18
Channels	16	8	16

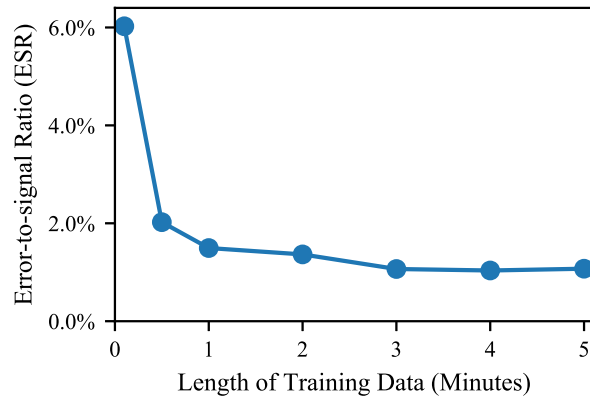


Figure 9: The validation energy-to-signal ratio (ESR) with different amounts of training data.

training a model. To assess the effect of the amount of training data, models were trained with different amounts of training data, and the effect on the validation loss was examined. The results are shown in Figure 9 for WaveNet3, the largest of the selected models. The results are averaged across the three modeled devices.

The validation loss decreases as the amount of training data is increased from 10 seconds to 3 minutes. Increasing the amount of training data past 3 minutes appears to have no significant effect on the validation loss.

## 6. RESULTS

Table 2 shows the ESRs of the selected models for the Tube Screamer, the DS-1 and the Big Muff. The reported ESR values were computed without pre-emphasis using the unseen test data set.

All selected models achieve a very small ESR on the Tube Screamer, suggesting that the proposed approach leads to a very accurate digital model. With the DS-1 and the Big Muff, the achieved ESR values are higher than with the Tube Screamer. In the tested configurations, the DS-1 and especially the Big Muff are highly nonlinear, due to their cascaded nonlinear stages. We believe that this explains the higher errors when compared to the Tube Screamer, which only has a single soft clipping stage. Overall, the WaveNet3 model has the lowest ESR and WaveNet1 has

Table 2: Error-to-signal ratio for selected test cases.

Model	TS7	DS-1	Big Muff
WaveNet1	0.069%	2.9%	9.9%
WaveNet2	0.050%	3.2%	7.1%
WaveNet3	<b>0.041%</b>	<b>2.1%</b>	<b>6.9%</b>



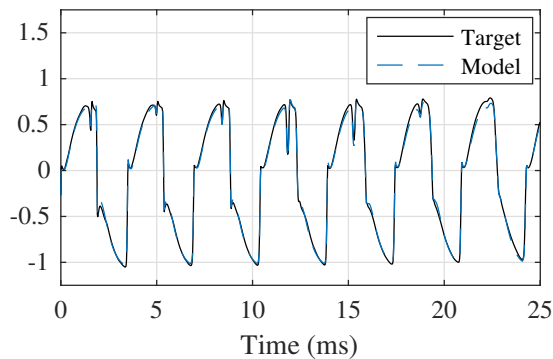


Figure 10: Waveforms of a guitar sound processed through the Boss DS-1, and through the WaveNet1 model.

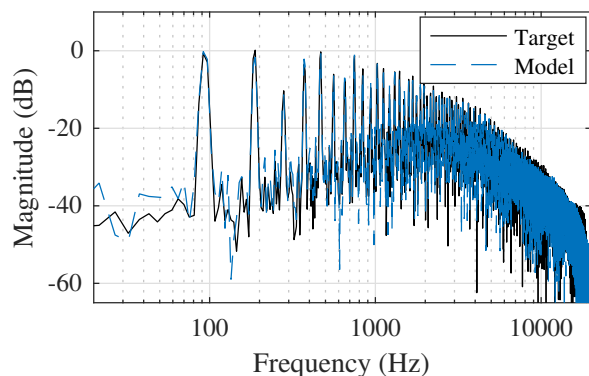


Figure 11: Spectra of a bass guitar sound processed through the Big Muff, and through the WaveNet3 model.

the highest ESR of the tested configurations. However, the differences between the models are relatively small.

The processing speeds of the models are shown in Table 3. The least accurate model (WaveNet1) runs fastest at 1.9 times faster than real time, whereas the most accurate model (WaveNet3) runs 1.1 times faster than real time.

Figure 10 shows the output waveform of the Boss DS-1 distortion effect to an electric guitar input signal from the test data set, and the corresponding output of the WaveNet1 model. The plot shows a good match between the target signal and the model prediction. Figure 11 shows the spectrum of a bass guitar signal processed through the Big Muff distortion effect, and the spectrum of the same signal processed through the WaveNet3 model. The spectrum of the model output matches the target spectrum well.

In order to estimate the aliasing introduced by the models, Figure 12 shows the spectrum of a 1245 Hz sinusoid fed through the WaveNet2 model of the Big Muff pedal. It appears that even though the models were trained with non-aliased data, the learned models suffer from aliasing. How-

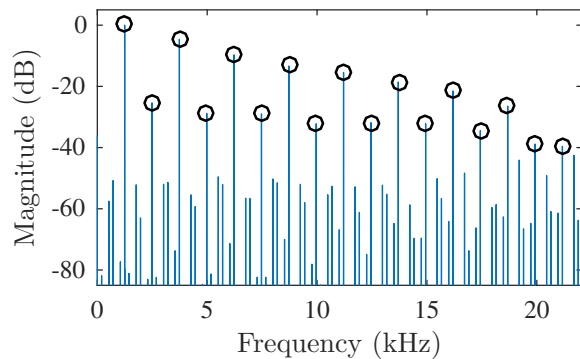


Figure 12: Spectrum of a 1245 Hz sinusoid fed through the WaveNet2 model of the Big Muff pedal. The black circles indicate the non-aliased components.

ever, while the aliasing is evident with a high-frequency sinusoidal input, no clear aliasing could be heard in the guitar and bass sounds processed through the models.

Several audio samples from all models are available online at the accompanying web page [27].

## 7. CONCLUSIONS

This work considered the use of deep neural networks for modeling of audio distortion effects. Three well-known guitar distortion pedals were modeled using a feedforward variant of the WaveNet neural network. Different model configurations were examined to find a suitable compromise between modeling accuracy and computational load. A real-time and low-latency implementation of the proposed deep neural network was developed. The results suggest that the proposed deep learning approach can be used to train accurate digital models of analog distortion effects, which can be run in real-time on a consumer-grade desktop computer. Future work will further study the aliasing behavior of neural network models.

## Acknowledgments

This work has been partially supported by the NordForsk Nordic University Hub “Nordic Sound and Music Computing Network – NordicSMC”, project number 86892. We acknowledge the computational resources provided by the Aalto Science-IT project.

## 8. REFERENCES

- [1] D. T. Yeh, J. Abel, and J. O. Smith, “Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations,” in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Bordeaux, France, Sept. 2007, pp. 197–204.
- [2] D. T. Yeh, J. S. Abel, A. Vladimirescu, and J. O. Smith, “Numerical methods for simulation of guitar distortion circuits,” *Computer Music J.*, vol. 32, no. 2, pp. 23–42, 2008.

Table 3: Processing speeds of the selected models reported as real-time (RT) factors. The fastest result is highlighted.

Model	WaveNet1	Wavenet2	WaveNet3
Speed ( $\times$ RT)	<b>1.9</b>	1.6	1.1

- [3] R. C. D. Paiva, S. D'Angelo, J. Pakarinen, and V. Välimäki, "Emulation of operational amplifiers and diodes in audio distortion circuits," *IEEE Trans. Circ. Syst. II: Express Briefs*, vol. 59, no. 10, pp. 688–692, Oct. 2012.
- [4] K. J. Werner, V. Nangia, A. Bernardini, J. O. Smith III, and A. Sarti, "An improved and generalized diode clipper model for wave digital filters," in *Proc. Audio Eng. Soc. 139th Conv.*, New York, NY, Oct. 2015.
- [5] J. Schattschneider and U. Zölzer, "Discrete-time models for non-linear audio systems," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Trondheim, Norway, Dec. 1999, pp. 45–48.
- [6] A. Novak, L. Simon, F. Kadlec, and P. Lotton, "Nonlinear system identification using exponential swept-sine signal," *IEEE Trans. Instr. Meas.*, vol. 59, no. 8, pp. 2220–2229, Aug. 2010.
- [7] C. Kemper, "Musical instrument with acoustic transducer," Aug. 2014, US Patent 8796530B2.
- [8] F. Eichas and U. Zölzer, "Black-box modeling of distortion circuits with block-oriented models," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Brno, Czech Republic, Sept. 2016, pp. 39–45.
- [9] —, "Gray-box modeling of guitar amplifiers," *J. Audio Eng. Soc.*, vol. 66, no. 12, pp. 1006–1015, Dec. 2018.
- [10] T. Hélie, "Volterra series and state transformation for real-time simulations of audio circuits including saturations: Application to the Moog ladder filter," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 747–759, May 2010.
- [11] S. Orcioni *et al.*, "Identification of Volterra models of tube audio devices using multiple-variance method," *J. Audio Eng. Soc.*, vol. 66, no. 10, pp. 823–838, Oct. 2018.
- [12] M. J. Kemp, "Analysis and simulation of non-linear audio processes using finite impulse responses derived at multiple impulse amplitudes," in *Proc. Audio Eng. Soc. 106th Conv.*, Munich, Germany, May 1999.
- [13] D. J. Gillespie and D. P. Ellis, "Modeling nonlinear circuits with linearized dynamical models via kernel regression," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust. (WASPAA)*, New Paltz, NY, USA, Oct. 2013, pp. 93–96.
- [14] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, "Deep learning for tube amplifier emulation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brighton, UK, May 2019.
- [15] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *ArXiv pre-print*, 2016, arXiv:1609.03499 [cs.SD].
- [16] ElectroSmash, "ElectroSmash – Electronics for audio circuits," Available online at: <https://www.electrosmash.com>, accessed: 2019-01-29.
- [17] Analog Man, "Ibanez Tube Screamer history," Available online at: <http://www.analogman.com/tshist.htm>, accessed: 2019-01-29.
- [18] D. T. Yeh, J. S. Abel, and J. O. Smith, "Simplified, physically-informed models of distortion and overdrive guitar effects pedals," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Bordeaux, France, Sept. 2007, pp. 10–14.
- [19] The Big Muff Pi Page, "Evolution of the Big Muff Pi circuit," Available online at: [http://www.bigmuffpage.com/Big\\_Muff\\_Pi\\_versions\\_schematics\\_part1.html](http://www.bigmuffpage.com/Big_Muff_Pi_versions_schematics_part1.html), accessed: 2019-03-26.
- [20] K. J. Werner, V. Nangia, J. O. Smith III, and J. S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Trondheim, Norway, Sept. 2015, pp. 387–394.
- [21] A. Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *Proc. Audio Eng. Soc. 108th Conv.*, Paris, France, Feb. 2000.
- [22] J. D. Markel and A. H. Gray, *Linear Prediction of Speech*. Berlin Heidelberg: Springer-Verlag, 1976.
- [23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learning Representations (ICLR)*, San Diego, CA, May 2015.
- [24] J. Abeßer, P. Kramer, C. Dittmar, G. Schuller, and I. Fraunhofer, "Parametric audio coding of bass guitar recordings using a tuned physical modeling algorithm," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Maynooth, Ireland, Sept. 2013, pp. 154–161.
- [25] C. Kehling, J. Abeßer, C. Dittmar, and G. Schuller, "Automatic tablature transcription of electric guitar recordings by estimation of score- and instrument-related parameters," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Erlangen, Germany, Sept. 2014, pp. 219–226.
- [26] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *Proc. 35th Int. Conf. Machine Learning*, Stockholm, Sweden, Jul. 2018.
- [27] E.-P. Damskägg, L. Juvela, and V. Välimäki, "Real-time modeling of audio distortion circuits with deep learning," accompanying web page, available online at: <http://research.spa.aalto.fi/publications/papers/smc19-black-box/>.

# MI-GEN~: AN EFFICIENT AND ACCESSIBLE MASS-INTERACTION SOUND SYNTHESIS TOOLBOX

**James Leonard**

Univ. Grenoble Alpes, CNRS, Grenoble INP\*  
GIPSA-Lab, 38000 Grenoble, France  
james.leonard@gipsa-lab.fr

**Jerome Villeneuve**

Univ. Grenoble Alpes, CNRS, Grenoble INP\*  
GIPSA-Lab, 38000 Grenoble, France  
jerome.villeneuve@gipsa-lab.fr

## ABSTRACT

Physical modelling techniques are now an essential part of digital sound synthesis, allowing for the creation of complex timbres through the simulation of virtual matter and expressive interaction with virtual vibrating bodies. However, placing these tools in the hands of the composer or musician has historically posed challenges in terms of a) the computational expense of most real-time physically based synthesis methods, b) the difficulty of implementing these methods into modular tools that allow for the intuitive design of virtual instruments, without expert physics and/or computing knowledge, and c) the generally limited access to such tools within popular software environments for musical creation. To this end, a set of open-source tools for designing and computing mass-interaction networks for physically-based sound synthesis is presented. The audio synthesis is performed within Max/MSP using the gen~ environment, allowing for simple model design, efficient calculation of systems containing single-sample feedback loops, as well as extensive real-time control of physical parameters and model attributes. Through a series of benchmark examples, we exemplify various virtual instruments and interaction designs.

## 1. INTRODUCTION

Over the last few decades, physically-based sound synthesis methods have evolved from computationally expensive & mostly non real-time techniques to one of the most active fields in Computer Music, now widely employed in digital sound synthesis, including in various commercialised solutions. Alongside historic methods, such as digital waveguides [1], modal synthesis [2] or lumped mass-interaction modelling [3], recent trends show an increasing number of methods rooted in mechanical and acoustical simulation, such as finite-difference-time-domain systems (FDTD) [4] or finite-element modelling [5]. Moreover, scattering techniques can be used to connect wave-based methods to Kirchhoff-based methods [6], enabling hybrid

modelling strategies and thus benefiting from the strengths of each individual paradigm.

### 1.1 Tools for physical modelling sound-synthesis

Advances in physical modelling and simulation techniques have, in most cases, been accompanied by the development of frameworks and tools allowing the afferent concepts to be manipulated. However, approaches differ significantly depending on the end goal:

- A number of tools, such as the Synthesis Toolkit [7] or Block Compiler [6] allow for expert users to design complex numerical simulations of musical instruments. They generally require a solid background in both computing and physics, and as such are targeted for the researcher or engineer rather than the musician or composer.
- Other tools hide some of the inner complexity of discrete-time physical models, offering more approachable concepts so that a composer or musician may be able to assimilate and incorporate them into a creative process, with little prior scientific knowledge. GENESIS [8] and MODALYS [2] are examples of such systems.

### 1.2 Modularity considerations

Another distinction between tools for creation by means of physical modelling is the degree of modularity that is made available to the user:

- Non-modular systems such as model-based digital pianos<sup>1</sup> [9], the Brass project [10] or Physical Audio's reverb plate<sup>2</sup> present the user with a fixed physical model to play, possibly allowing him/her to manipulate physical parameters or chose between several modes of excitation, etc.
- Semi-modular systems generally propose a base of common physical structures (strings, plates, etc.) available as primitives that can be assembled together to create virtual objects [2, 11, 12].
- Entirely-modular systems such as GENESIS [8] allow the user to design virtual structures "from the ground up" by assembling basic physical elements (masses, springs, non-linear interactions).

\*Institute of Engineering, Univ. Grenoble Alpes.

Copyright: © 2019 James Leonard et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> [www.pianoteq.com](http://www.pianoteq.com)

<sup>2</sup> <https://physicalaudio.co.uk/PA1>

We note that as methods such as finite-difference time-domain schemes evolve, offering evermore realistic synthesised sound, the inherent complexity of such models renders intuitive modification of the instruments difficult for non-expert users; higher-level tools are generally limited to a semi-modular approach [13] by connecting plates, strings or bars through locally handled lumped interactions.

Systems aiming for a more fundamental, constructivist, type of modularity must strike a compromise by leaning on physical formulations that can be split into fundamental “atomic” elements: such is the case of mass-interaction physical modelling.

### 1.3 Modular physical modelling for artistic creation

#### 1.3.1 (Unsupervised!) modelling

The study of mass-interaction modelling for musical creation is largely documented, essentially through work conducted since the early 80’s at ACROE [3, 8]. One of the key features of this approach - and an essential distinction from most other physical modelling methods - is that virtual physical objects can be designed intuitively (or empirically) by assembling basic physical elements into a network.

In this case, “modelling” does not necessarily refer to creating a virtual simulation that presents similar behaviour to that of an existing physical instrument (see [14]); rather, modelling is the activity of exploring virtual mechanical constructions in order to discover behaviours and sounds that are judged interesting (regardless of any criteria of realism). Emphasis is consequently placed on providing various tools for the construction, generation and analysis of virtual physical networks in order to assist the composer or musician during the design process [15, 16].

#### 1.3.2 Towards open tools for mass-interaction modelling

Recent years show a regain of interest for modular and more accessible physically-based synthesis methods such as mass-interaction modelling, stimulated partly by the possibilities of force-feedback interaction with such models [17, 18].

Efforts have been made to provide accessible open-source tools for environments such as Max/MSP. The *HSP* project [19] presents masses and interactions directly as Max objects, that can be interconnected to form a network. However, the single sample delay (cf. Section 2) needed for the position/force feedback loop imposes that Max runs with a *VectorSize* of 1, hindering the performance of general audio workflow in order to preserve the integrity and stability of the discrete physical computation.

*Synth-A-Modeler* [20] is based on the Faust compiler<sup>3</sup> and allows creating and compiling mass-interaction models (and also hybrid models with waveguide and modal synthesis) for a variety of targets (PureData, Max, standalone, etc.). Compiled models are then available as black-box objects that can be acted upon and observed through inlets and outlets.

<sup>3</sup> Recent work on the formalisation of mass-interaction networks in FAUST was also conducted by the authors [21].

#### 1.3.3 Presented work: *mi-gen~*

By leveraging the properties of the *gen~* system, the work presented in this paper allows for efficient implementation of mass-interaction models within Max/MSP. These models can be generated using a dedicated scripting tool, or coded directly within *gen~*’s *codebox* object, leaving the user free to modify them at any time. Large scale models can be designed (with guaranteed single-sample loopback), visualised, interacted with, and integrated into complex workflows within Max.

First, we provide a brief introduction to algorithmic and computational aspects of mass-interaction physical modelling. Then, we present how such models can be coded and computed with the *mi-gen~* toolbox. Examples and system performance are then discussed, with perspectives for future work.

## 2. MASS-INTERACTION PHYSICAL MODELLING BASICS

As the name suggests, mass-interaction models are composed on the one side of *mass-type* elements, and on the other *interaction-type* elements. The modularity of the formalism stems from the fact that the basic equations for each are discretised individually, and can then be assembled freely by following a small number of connection rules [3, 22, 23]. The principles of mass-interaction modelling can operate on position and force data of any spatial dimension (scalar values, 2D or 3D vectors, etc.). The equations below and the implementation in this paper are 1D, meaning that all masses move along a single axis.

A discrete-time implementation of a punctual mass is obtained by applying the second order central difference scheme to Newton’s second law ( $f$  is the force applied to the mass,  $m$  is its inertia,  $a$  its acceleration, and  $x$  its position):

$$f = ma = m \frac{d^2 x}{dt^2} \quad (1)$$

Resulting in the following normalised form (with discrete-time positions and forces noted  $X$  and  $F$ ):

$$X_{(n+1)} = 2X_{(n)} - X_{(n-1)} + \frac{F_{(n)}}{M} \quad (2)$$

With  $M$ , the discrete time inertial parameter defined as  $M = m/\Delta T^2$  (where  $\Delta T$  is the sampling interval).

As an example of a simple interaction element, the viscoelastic force applied by a linear spring (with stiffness coefficient  $k$ , damping coefficient  $z$  and resting length of  $l_0 = 0$ ) connecting a mass  $m_2$  at the position  $x_2$  to a mass  $m_1$  at the position  $x_1$  is given by:

$$f_{1 \rightarrow 2} = -k(x_2 - x_1) - z \frac{d(x_2 - x_1)}{dt} \quad (3)$$

Approximating the velocity by applying the backward Euler difference scheme, we obtain:

$$\begin{aligned} F_{(n)} = & -K(X_{2(n)} - X_{1(n)}) \\ & - Z((X_{2(n)} - X_{2(n-1)}) \\ & - (X_{1(n)} - X_{1(n-1)})) \end{aligned} \quad (4)$$



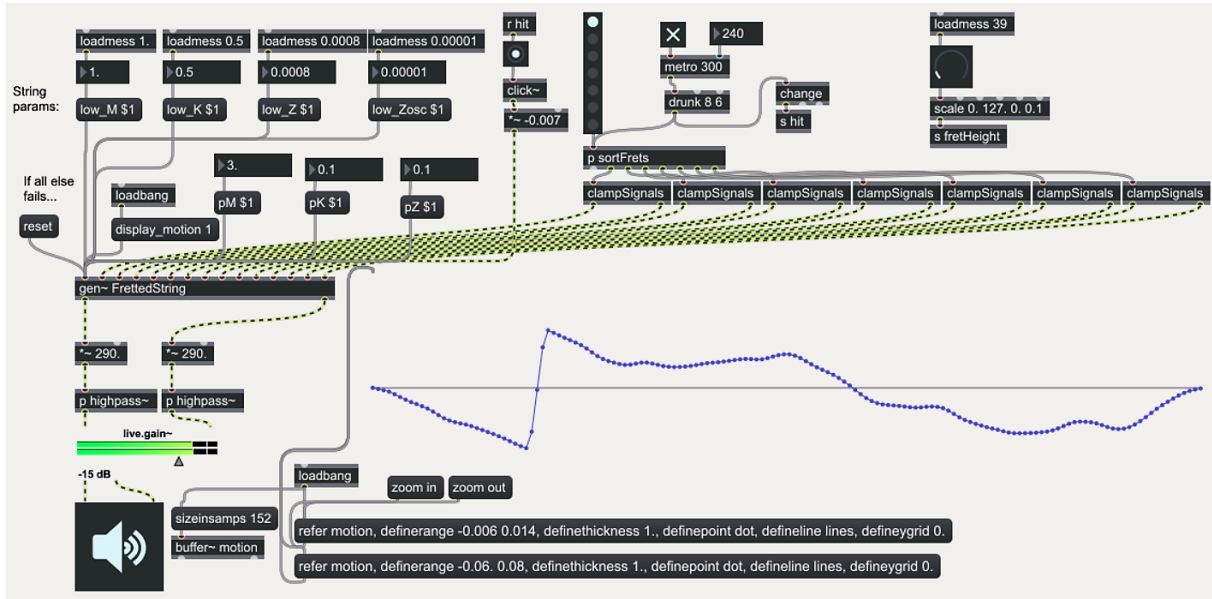


Figure 1. Screenshot of a Max patch implementing a string model. Control parameters are sent as messages to the `gen~` patch, and external position and force inputs are provided as audio-rate signals.

With the discrete-time stiffness parameter  $K = k$ , and the discrete-time damping parameter  $Z = z/\Delta T$ . The resulting force is applied symmetrically to each mass (Newton’s third law):

$$\begin{aligned} F_{1(n)} - &= F_{(n)} \\ F_{2(n)} + &= F_{(n)} \end{aligned} \quad (5)$$

Each interaction acting upon a mass effectively sums its calculated force into a buffer, which equates (after weighting by  $M$ ) to the total acceleration applied to the mass during this time-step. The computation of a mass-interaction network consists in a closed-loop computation of the masses and of the interactions. By convention, we will say that the vibration of 1D topological structures occurs along the  $z$ -axis.

### 3. THE MI-GEN~ LIBRARY

The `mi-gen~` toolbox is an open-source Max package, available on GitHub<sup>4</sup>. It is composed of several documentation and example patches, as well as a library of functions for `gen~`’s `codebox` object. `Codebox` is a textual coding environment within `gen~` (based on a syntax similar to C), giving access to all `gen~` specific functions such as *History*, *Delay*, *Buffer/Data* access, etc.

As in all `gen~` patches, feedback paths with a single sample of delay are possible regardless of Max’s *VectorSize*, which is crucial for implementing the closed loop calculations of equations (2) and (4) for all of the masses and interactions in the physical model. Figure 1 shows a Max patch centred around a “fretted string” `gen~` physical model.

<sup>4</sup> <https://github.com/mi-creative/mi-gen>

### 3.1 Code structure

The sequence in Figure 2 shows the code structure of a simple model: a harmonic oscillator (a mass, connected to a fixed point by a dampened spring) that can be subjected a force input via the first inlet of the `codebox` object, and can be struck using a contact interaction (with stiffness, damping and threshold distance parameters) by another mass whose position is controlled by the second inlet.

#### 3.1.1 Physical variables and initial states

Material points of the model (including “avatars” for external position inputs) are stored as *Data* objects (equivalent to an array of floating point 64-bit values). Each one is composed of three values, corresponding respectively to the point’s current position, previous position, and accumulated force buffer. These attributes are set to initial conditions during an initialisation phase. By convention, modules are set with an initial position and velocity (inferred by the previous position), and with null acceleration.

Interactions have no physical variables other than their own parameters, since an interaction acts directly (and symmetrically) on the *Data* structures of the two masses that it connects.

#### 3.1.2 Computational loop

The simulation of the model’s dynamics is performed by computing all of the mass-type algorithms then all of the interaction type algorithms, for each time step (each audio sample in our case). All of the physical algorithms are implemented inside the *migen-lib.genexpr* file.

External position inputs are applied to the “avatar” modules during the mass-phase, and external force inputs are applied to corresponding masses during the interaction-phase. Position and/or force values from “observed” masses are collected at the end of the cycle and routed to outputs.

```

require("migen-lib");

// Model data structures
Data m_in2(3);
Data gnd(3);
Data m1(3);

// Control Rate Parameters
Param Z(0.0001);
Param K(0.01);
Param M(1.);

History model_init(0);
// Model init phase
if(model_init == 0){
    init_mat(m_in2, 1, 1);
    init_mat(gnd, 0, 0);
    init_mat(m1, 0, 0);
    model_init = 1;
}

// Model computation
update_input_pos(m_in2, in2);
compute_ground(gnd);
compute_mass(m1, M);
compute_contact(m1, m_in2, 0.1, 0, 0);
apply_input_force(m1, in1);
compute_spring_damper(m1, gnd, K, Z);

out1 = get_pos(m1);

```

Figure 2. A simple mass-interaction model expressed in *genexpr* code, using the *mi-gen~* library.

### 3.1.3 Parameters of the physical algorithms

The physical parameters expressed inside *mi-gen~* are *normalised discrete-time* parameters as defined in Section 2. On the one hand, this renders the behaviour of models dependent on the sample rate for a given set of parameters, on the other it provides a direct view of stability conditions, defined at each mass as  $4M > K + 2Z$  (with  $M$  the inertia of the mass, and  $K$  and  $Z$  the summed stiffness and damping applied to this mass by interactions - see §7). Translations to and from standard unit parameters can easily be established allowing the user to manipulate either one.

Parameters may be:

- hard-coded values that are immutable once the patch is compiled,
- control-rate parameters that can be dynamically modified by sending messages to the *gen~* object,
- audio-rate parameters, added as signal inputs to the *codebox* patch.

The latter is preferable for fast-varying parameters [24], for instance in cases such as dynamically re-tuning the pitch of a model according to input MIDI notes.

## 3.2 The MIMS scripting system

Although creating physical models directly in *codebox* is fairly intuitive, designing larger scale objects is much easier

using higher level tools to describe the topology. MIMS<sup>5</sup> (Mass-Interaction Model Scripter) is a basic editor written in Python for this purpose, providing compilation into dsp code for either *gen~* or FAUST [21].

Models are described in a format similar to PNSL [16]: each physical element is given a unique identifier, or label (e.g. *@mass1*), that can be referenced by other elements (e.g. when connecting an interaction between two masses). For instance:

**@spr springDamper @m1 @m2 0.1 0.001**

creates a dampened spring connecting masses *m1* and *m2* with a discrete-time stiffness of 0.1 and discrete-time damping of 0.001. The inputs and outputs of the *gen~* patch are also based on labels:

**@in1 frcInput @m1  
@out2 posOutput @m2**

routes a force signal from the inlet *in1* to the mass *m2* and routes an observation of *m2*'s position to the outlet *out2*.

Parameters also function as labels and can be used in place of hard-coded values in the definition of any module's parameters as follows:

**@M param 1.  
@K param 0.01  
@Z param 0.001  
...  
@cel osc M K Z 0. 0.1**

The above code creates labelled inertia, stiffness and damping control-rate parameters and then creates an integrated harmonic oscillator with these parameters, set at initial position  $z = 0$  with an initial velocity of 0.1 metres-per-sample. Audio-rate parameter inputs are automatically placed after any explicitly defined inputs for the patcher.

MIMS also provides rudimentary functions for automated generation of larger scale topological structures, such as strings, rectangular, triangular and hexagonal membranes.

## 4. REAL-TIME VISUALISATION

### 4.1 Visualising deformations using Motion Buffers

So far, we have only been able to observe points of a physical model that we route to outlets at the audio rate. However, understanding the behaviour of a mass-interaction model is greatly facilitated by visualising the deformations of the entire object, even if we only "listen" to it in a few points. This is achieved by creating *motion buffers*. Motion buffers are simple MSP *buffer~* objects, used a little unconventionally: instead of storing a temporal waveform, the buffer stores an instantaneous snapshot of the positions of a set of masses. MIMS provides two uses for motion buffers:

- A 1-channel generic buffer, containing the motion of all the mass-type elements in the model along the *z* axis (in order of creation).

<sup>5</sup> <https://github.com/mi-creative/MIMS>

Model Name	Nb. Masses	Nb. Int.	Comp. Time.	CPU Load
1000 Mass String	1000	1002	$\infty$	-
Mesh (25x20)	506	965	3 min	67%
Mesh (15x15)	225	430	22 s	15%
Fretted String	155	203	7.7 s	8%
Bowed String	152	158	5.8 s	7%
Plucked Harmonics	152	158	5.4 s	6%
Drunk Triangles*	4	5	<1 s	$\approx$ 16%

Table 1. Benchmarking: Number of masses and interactions, compilation time, and Max/MSP CPU load. Measurements were made on a Dell Precision 5530 running Windows 10 & Max 7.3.5. Specs: Intel i7-8850H 4 cores at 2.6GHz, 16GB RAM, 44.1kHz sampling rate, buffer size & vector size of 256 samples. \* Drunk Triangles is a small model instantiated dynamically with up to 100 voices, using poly~.

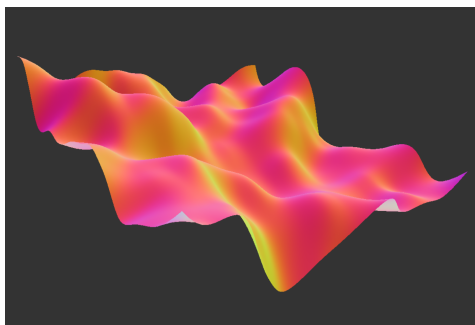


Figure 3. Vibratory deformations of a mesh attached at each corner. Underlying physical model: 20x20 masses. NURBS-based rendering in Jitter.

- Specific 3-channel buffers, containing the motion of a set of mass-type elements in the model along the  $z$  axis, as well as fixed  $x$  and  $y$  coordinates. Masses are added to buffers by adding the buffer name and extra  $x$ - $y$  coordinates after the standard module parameters.

The former allows quick visualisation of the model state by drawing the motion buffer with the plot~ object. The latter is useful for creating visual arrangements that correspond to the topological nature of a mass-interaction network, e.g. grid-based distributions along  $x$  and  $y$  according to generated mesh patterns, which allows to represent modal deformations and wave propagation along the matter.

Inside the gen~ patch, the motion buffers are refreshed with new positions once every 200 audio steps (a rate of 220.5 Hz for a sampling rate of 44.1kHz). This limits the computational costs of writing the data, as it is only used for visualisation purposes.

## 4.2 Rendering techniques

Within Max, Jitter offers powerful tools for visual rendering, including surface rendering algorithms such as *Non-Uniform, Rational, B splines* (NURBS). This technique is used to render smooth curves and surfaces from a limited number of control points (in our case, the  $x$ - $y$ - $z$  positions of the masses of a physical model). Using specific motion buffers as discussed above, simple model topologies such as strings and rectangular membranes can be rendered as shown in Figure 3.

## 5. BENCHMARKING & EXPERIMENTATION

The mi-gen~ toolbox provides a set of tutorial and example patches<sup>6</sup>, showing a hands-on approach to designing models and control/interaction strategies within Max/MSP.

### 5.1 Performance benchmark

Table 1 shows the results of a selection of models (mostly examples from the toolbox), in terms of complexity, compile-time of the gen patcher, and CPU usage within Max.

Results show that the main limitation for large scale models stems from the gen compilation phase: models with approx. 650 modules take over 20 seconds to compile, but only occupy around 15 % of the CPU when running. The load displayed by the Max/MSP monitor is reasonable for all models that pass the compilation phase, and shows no noticeable difference whether the visualisation of motion data is active or not.

Beyond a certain volume of codebox code, the compilation of the patch hangs (cf. the 1000 Mass String in Table 1). Reasons for this limitation will be investigated. It is worth noting that the authors experienced similar issues with the Faust compiler [21].

Performance of complex models can be optimised by means of dynamically allocated voices using the poly~ system, as is the case in the *Drunk Triangles* example, which contains up to 100 instances of a simple mass dropping and bouncing on a “triangle” resonator (shown in Figure 4). Each instance is tuned according to a pseudo random sequence generated with Max’s *drunk* object, and frees its voice when it detects that the mass has stopped bouncing. Figure 5 shows a visual representation of the entire instrument.

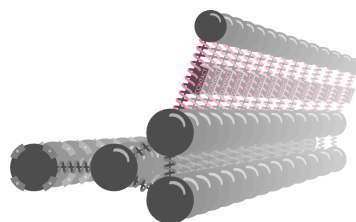


Figure 4. Schematic representation of the *Drunk Triangles* model.

<sup>6</sup> Video demonstrations are provided at: [mi-creative.eu/tool\\_migen.html](http://mi-creative.eu/tool_migen.html)

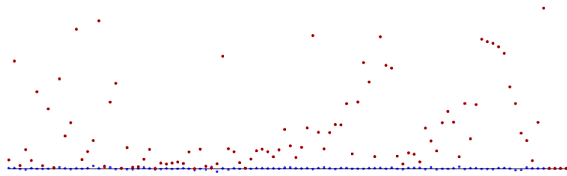


Figure 5. Visual representation of the *Drunk Triangles* polyphonic instrument: bouncing masses in red, resonating structures in blue.

## 5.2 The fretted string model

Plucked or struck string instruments have long been a topic of interest in physical modelling, from the acoustical properties of the string itself [25] to the non-linear collisions [26, 27] that occur in fretted instruments. Below we present a simple mass-interaction implementation of such a model.

### 5.2.1 Fretting mechanisms in 1D space

Previous work by the authors [28] presented fretted string mass-interaction models with an analogy to the guitar fret-board, where certain string masses are pinned down against fixed points, set underneath them. However, this poses problems since, unlike in real life where fretting gestures are orthogonal to strumming ones, in the 1D virtual model the string is excited in the same axis as the fretting gesture. This forces frets to be placed very low and distributed unevenly beneath the string in order to avoid fret buzz.

A more functional analogy in our case comes from the tuning forks found on concert harps. When rotated by operating the instrument's pedals, they apply pressure in both directions simultaneously, pinning the string without causing excess displacement. Within Max, it is straightforward to setup two opposite position signals that operate the tuning fork "clamp", and to replicate the mechanism in as many points as desired.

### 5.2.2 Model description and behaviour

The vibrating model is a simple string attached between two fixed points. An external force input triggers a mass that moves downwards and collides with the string. Multiple external position inputs control seven frets (or clamps), positioned at intervals corresponding to the diatonic scale. The height of the frets and the speed of the clamping mechanisms can be modified in real-time: we can choose to completely avoid any fret-buzz, or dial in just enough for things to sound lively and interesting... or even go crazy with extremely rattly down-tuned strings! The Max patch is shown in Figure 1.



Figure 6. Schematic representation of the fretted string model (clamping mechanisms on the left side).

### 5.2.3 "Physical Interaction" Realism

Once again, we stress that the created model does not strive for any true acoustical realism: indeed, the string's masses are limited to transverse motion, it is modelled with linear springs, has no vibrating body, the discrete number of masses imposes slightly off-key fret positions... the list goes on. However, even in this extremely reductive model, careful physical interaction design brings forth perceptually important emergent non-linear behaviour, inferring character and unpredictability to the instrument.

We could say that the mass-interaction paradigm yields *physical interaction realism*, in the sense that it faithfully represents anything that can be represented with Newtonian point-based mechanics, and that any path the user chooses to follow from there on - be it searching to reproduce real-life phenomena or exploring the unknown - is entirely up to him.

## 6. DISCUSSION

This paper has presented mi-gen~, a new library and set of tools for mass-interaction physical modelling sound synthesis in Max/MSP. In comparison to existing tools, it offers efficient computation entirely within Max, without precompiling the models into static black boxes. This allows for on-the-fly iterative model design, direct access to physical parameters and to the model state (which can be visualised using any method seen fit) and - maybe most importantly - it empowers the user by providing a hands-on programming framework for 1D mass-interaction physics, with vast possibilities for customisation. As such, it constitutes a new experimentation ground for combining mass-interaction physics with the immense panoply of signal-based tools available inside of Max, for any sound synthesis, transformation or analysis purpose.

An immediate perspective of this work is integration with affordable force feedback systems (such as the FireFader or the Haply<sup>7</sup> device). A more fundamental perspective for musical creation is considering sound synthesis with 3D mass-interaction models, as modelling virtual objects with spatial attributes naturally addresses many of the limitations of 1D modelling, in particular regarding non-linear dynamics.

It is our hope that this work will constitute another step towards opening mass-interaction modelling to a larger spectrum of users, in the Computer Music community and beyond. Despite the apparent simplicity of the formalism, we are convinced that much has still to be said, discovered, and experimented.

## 7. APPENDIX: STABILITY CONDITIONS FOR THE HARMONIC OSCILLATOR

Following equations (2) and (4), the discrete time recurrence of the linear harmonic oscillator composed of a mass, dampened spring and a fixed point can be expressed (in the absence of external forces other than the spring) as:

<sup>7</sup> <http://haply.co>



$$X_{(n+1)} + \left(\frac{K+Z}{M} - 2\right) \cdot X_{(n)} + \left(1 - \frac{Z}{M}\right) \cdot X_{(n-1)} = 0 \quad (6)$$

The associated characteristic polynomial is:

$$r^2 + A \cdot r + B = 0 \quad (7)$$

with  $A = \left(\frac{K+Z}{M} - 2\right)$  and  $B = \left(1 - \frac{Z}{M}\right)$ , giving the discriminant  $\Delta = A^2 - 4B$ .

### 7.1 Case 1: Oscillating conditions ( $\Delta < 0$ )

If  $\Delta < 0$  equation (7) possesses complex roots and gives an oscillating solution. We can therefore express the physical parameters that bound oscillating solutions as:

$$\left(\frac{K+Z}{M} - 2\right)^2 - 4\left(1 - \frac{Z}{M}\right) < 0 \quad (8)$$

Which can be developed into:

$$(K + Z)^2 < 4KM \quad (9)$$

The pseudo-periodic resulting oscillator is of the form:

$$X_{(n)} = \chi \rho^n \cos(n\omega_p + \varphi) \quad (10)$$

With  $\rho = \sqrt{B}$  and  $\omega_p = \arccos\left(-\frac{A}{2\sqrt{B}}\right)$ . Oscillations converge towards 0 for  $0 < \frac{Z}{M} < 1$  and are divergent otherwise.

### 7.2 Case 2: Non-oscillating conditions ( $\Delta \geq 0$ )

$\Delta = 0$  leads to a single real root and a solution of the form:

$$\begin{aligned} X_{(n)} &= (\alpha + \beta n) \left(-\frac{A}{2}\right)^n \\ &= (\alpha + \beta n) \left(1 - \frac{K+Z}{2M}\right)^n \end{aligned} \quad (11)$$

This condition equates to critical damping in the oscillator:

$$Z = 2\sqrt{KM} - K \quad (12)$$

This solution converges if  $|1 - \sqrt{\frac{K}{M}}| < 1$ , resulting in the following stability limits:

$$0 < \sqrt{\frac{K}{M}} < 2 \quad (13)$$

If  $\Delta > 0$ , equation (7) has two real roots  $r_1$  and  $r_2$  and the solution is of the form:

$$X_{(n)} = \alpha(r_1)^n + \beta(r_2)^n \quad (14)$$

with roots:

$$r_{1,2} = \frac{-A \pm \sqrt{\Delta}}{2} \quad (15)$$

$X_{(n)}$  converges towards zero if  $|r_1| < 1$  and  $|r_2| < 1$ . Taking  $|\frac{-A+\sqrt{\Delta}}{2}| < 1$  we can express two inequalities:

$$\begin{aligned} \sqrt{\Delta} &< 2 - A \\ \sqrt{\Delta} &< 2 + A \end{aligned} \quad (16)$$

Which after developing  $\Delta$  and squaring the inequalities (assuming  $0 < K+Z < 4M$  so that both sides are positive) results in:

$$\begin{aligned} A - B &< 1 \\ A + B &> -1 \end{aligned} \quad (17)$$

Leading to the final stability conditions as functions of  $M$ ,  $K$  and  $Z$ :

$$\begin{aligned} \frac{K}{M} &> 0 \\ K + 2Z &< 4M \end{aligned} \quad (18)$$

### 7.3 Final stability conditions

The stability conditions and regimes for our mass-interaction harmonic oscillator are given in Figure 7. The oscillator is numerically stable if the stiffness, damping and mass parameters verify:

$$\begin{aligned} K + 2Z &< 4M \\ K/M &> 0 \end{aligned}$$

Additionally the system will be in an oscillatory regime if the same parameters verify:

$$\begin{aligned} (K + Z)^2 &< 4KM \\ 0 < Z/M < 1 \end{aligned}$$

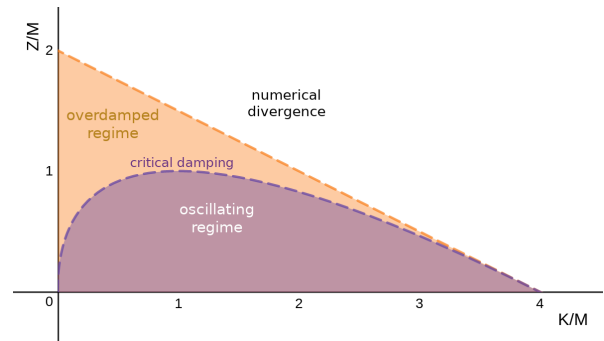


Figure 7. Stability conditions for the harmonic oscillator.

Generally speaking, the static regime stability condition for a mass connected to any number of linear springs and/or dampers can be expressed by analysing the harmonic oscillator with  $K_{eq} = \sum K$  and  $Z_{eq} = \sum Z$ .

## 8. REFERENCES

- [1] J. O. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, no. 4, pp. 74–91, Winter 1992.
- [2] J.-M. Adrien, "The missing link: Modal synthesis," in *Representations of musical signals*. MIT Press, 1991, pp. 269–298.
- [3] C. Cadoz, A. Luciani, and J. L. Florens, "Cordis-anima: a modeling and simulation system for sound and image synthesis: the general formalism," *Computer music journal*, vol. 17, no. 1, pp. 19–29, 1993.

- [4] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. Chichester, UK: John Wiley and Sons, 2009.
- [5] R. Panneton and N. Atalla, "An efficient finite element scheme for solving the three-dimensional poroelasticity problem in acoustics," *The Journal of the Acoustical Society of America*, vol. 101, no. 6, pp. 3287–3298, 1997.
- [6] R. Rabenstein, S. Petrausch, A. Sarti, G. De Sanctis, C. Erkut, and M. Karjalainen, "Blocked-based physical modeling for digital sound synthesis," *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 42–54, 2007.
- [7] P. Cook and G. Scavone, "The Synthesis Toolkit (stk)," in *Proceedings of the International Computer Music Conference (ICMC-99)*, Beijing, China, 1999.
- [8] N. Castagné and C. Cadoz, "Genesis: a friendly musician-oriented environment for mass-interaction physical modeling," in *ICMC 2002-International Computer Music Conference*. MPublishing, 2002, pp. 330–337.
- [9] B. Bank and J. Chabassier, "Model-based digital pianos: from physics to sound synthesis," *IEEE Signal Processing Magazine*, 2018.
- [10] C. Vergez and P. Tisserand, "The brass project, from physical models to virtual musical instruments: Playability issues," in *International Symposium on Computer Music Modeling and Retrieval*. Springer, 2005, pp. 24–33.
- [11] S. Bilbao, "A modular percussion synthesis environment," in *Proc. of the 12th Int. Conference on Digital Audio Effects (DAFx-09)*, 2009.
- [12] S. Bilbao, A. Torin, P. Graham, J. Perry, and G. Delap, "Modular physical modeling synthesis environments on gpu," in *ICMC*, 2014.
- [13] C. J. Webb and S. Bilbao, "On the limits of real-time physical modelling synthesis with a modular environment," in *Proceedings of the International Conference on Digital Audio Effects*, 2015, p. 65.
- [14] V. Välimäki and T. Takala, "Virtual musical instruments. natural sound using physical models," *Organised Sound*, vol. 1, no. 2, pp. 75–86, 1996.
- [15] J. Villeneuve, C. Cadoz, and N. Castagné, "Visual representation in genesis as a tool for physical modeling, sound synthesis and musical composition," in *New Interfaces for Musical Expression 2015*, 2015, pp. 195–200.
- [16] J. Villeneuve and C. Cadoz, "Understanding and tuning mass-interaction networks through their modal representation," in *40th International Computer Music Conference/11th Sound and Music Computing Conference*, 2014, pp. 1490–1496.
- [17] E. Berdahl, "An introduction to the Synth-A-Modeler compiler: Modular and open-source sound synthesis using physical models," in *Proceedings of the Linux Audio Conference (LAC-12)*, Stanford, USA, May 2012.
- [18] J. Leonard, N. Castagné, C. Cadoz, and A. Luciani, *The MSCI Platform: A Framework for the Design and Simulation of Multisensory Virtual Musical Instruments*. Cham: Springer International Publishing, 2018, pp. 151–169.
- [19] D. Overholt, A. Kontogeorgakopoulos, and E. Berdahl, "Hsp v2: Haptic signal processing with extensions for physical modeling," *Haptic Audio and Interaction Design 2010 Program and Papers*, pp. 61–62, 2010.
- [20] E. Berdahl and A. Kontogeorgakopoulos, "The fire-fader: Simple, open-source, and reconfigurable haptic force feedback for musicians," *Computer Music Journal*, vol. 37, no. 1, pp. 23–34, 2013.
- [21] J. Leonard, J. Villeneuve, R. Michon, S. Letz, and Y. Or-larey, "Formalizing mass-interaction physical modeling in FAUST," in *Linux Audio Conference (LAC'19)*. Stanford University, USA, 2019.
- [22] E. Incerti and C. Cadoz, "Modélisations et simulations de structures vibrantes en CORDIS. Matériaux et paramètres pour la création musicale," in *Deuxièmes Journées d'Informatique musicale*, Paris, France, 1995, pp. 173–183.
- [23] A. Kontogeorgakopoulos and C. Cadoz, "Cordis anima physical modeling and simulation system analysis," in *4th Sound and Music Computing Conference 2007*. National and Kapodistrian University of Athens, 2007, pp. 275–282.
- [24] E. Berdahl, "Audio-rate modulation of physical model parameters," in *ICMC*, 2014.
- [25] T. Tolonen, V. Valimäki, and M. Karjalainen, "Modeling of tension modulation nonlinearity in plucked strings," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 3, pp. 300–310, 2000.
- [26] S. Bilbao, A. Torin, and V. Chatziioannou, "Numerical modeling of collisions in musical instruments," *Acta Acustica united with Acustica*, vol. 101, no. 1, pp. 155–173, 2015.
- [27] G. Evangelista and F. Eckerholm, "Player–instrument interaction models for digital waveguide synthesis of guitar: Touch and collisions," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 4, pp. 822–832, 2010.
- [28] J. Leonard and C. Cadoz, "Physical modelling concepts for a collection of multisensory virtual musical instruments," in *Proceedings of the Conference on New Interfaces for Musical (NIME15)*, Baton Rouge, USA, May 2015.

# COMBINING TEXTURE-DERIVED VIBROTACTILE FEEDBACK, CONCATENATIVE SYNTHESIS AND PHOTOGRAMMETRY FOR VIRTUAL REALITY RENDERING

**Eduardo Magalhães**  
Universidade do Porto  
eduardom@fe.up.pt

**Emil Rosenlund Høeg**  
Aalborg University  
erh@create.aau.dk

**Gilberto Bernardes**  
Universidade do Porto  
gba@fe.up.pt

**Jon Ram Bruun-Pedersen**  
Aalborg University  
jpe@create.aau.dk

**Stefania Serafin**  
Aalborg University  
sts@create.aau.dk

**Rolf Nordahl**  
Aalborg University  
rn@create.aau.dk

## ABSTRACT

This paper describes a novel framework for real-time sonification of surface textures in virtual reality (VR), aimed towards realistically representing the experience of driving over a virtual surface. A combination of capturing techniques of real-world surfaces are used for mapping 3D geometry, texture maps or auditory attributes (aural and vibrotactile) feedback. For the sonification rendering, we propose the use of information from primarily graphical texture features, to define target units in concatenative sound synthesis. To foster models that go beyond current generation of simple sound textures (e.g., wind, rain, fire), towards highly “synchronized” and expressive scenarios, our contribution draws a framework for higher-level modeling of a bicycle’s kinematic rolling on ground contact, with enhanced perceptual symbiosis between auditory, visual and vibrotactile stimuli. We scanned two surfaces represented as texture maps, consisting of different features, morphology and matching navigation. We define target trajectories in a 2-dimensional audio feature space, according to a temporal model and morphological attributes of the surfaces. This synthesis method serves two purposes: a real-time auditory feedback, and vibrotactile feedback induced through playing back the concatenated sound samples using a vibrotactile inducer speaker.

## 1. INTRODUCTION

Contact between interacting objects in a natural environment often conveys information about the objects themselves. For instance, when walking on a surface, the contact between our feet and the ground will provide auditory and haptic feedback, which in many cases may be sufficient for us to recognize the surface we are walking on [1]. For initial surface identification, humans may utilize vision [2], but studies have shown how the multimodal perception of surfaces contributes to our experience and recog-

nition of surfaces [3]. If vision does not convey sufficient surface information, we may be able to discern it through other sensory channels. An example would be when gradually investigating a frozen lake, to test for safety. Simply looking at the lake will likely not provide sufficient information, so we gradually test from feedback; tapping or stepping on the lake to gauge its thickness from the multisensory feedback it causes. How does it sound? How does it feel? Does the stepping even produce visual feedback, from changes such as cracks or any other visible reactions to weight or impact? However, there seems to be no definitive consensus on what constitutes a texture [4], but perception of surface textures can be considered multidimensional (rough/smooth, fine/rough, slippery/resistant, etc.) as well as multisensory (experienced through haptics, vision and audition) [2]. As such, we expect surface interaction feedback, whether we need it to explore a surface (e.g. ice thickness) or simply expect it to be part of our contact experience in natural environments.

To our knowledge, most solutions to aural and vibrotactile feedback in virtual reality (VR) rely on static, fixed or synthetic solutions, triggered by binary states (i.e., on and off) or random variation. While these methods have been assessed as expressive and natural, in light of the degree they play within complex multimodal scenarios, no systematic evaluation has addressed the degrees of correspondence between modalities at finer temporal granularities [5]. In this context, our work strives for a method capable of generating audio streams with highly controllable nuances for aural and vibrotactile feedback using concatenative sound synthesis (CSS) [6]. Despite the lack of use-cases adopting this sample-based synthesis in VR, the technique has proven to be quite robust in generating dynamic, evolving and ever-changing sound textures from short audio excerpts. Additionally, it enables the creation of audio streams, at different temporal granularities.

The integration of CSS in VR can tackle two important limitations of the technique, as identified in [6]: 1) the evaluation of a descriptors’ salience, notably the difference between the aural and the vibrotactile descriptor spaces, and 2) the definition of targets which convey both the finer degree of user-controllable actions interacting with the Virtual Environment (VE). Ultimately, we aim to foster a unified aural and vibrotactile framework, which stresses a

Copyright: © 2019 Eduardo Magalhães et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

high level of control, adaptation and presumably a greater sense of immersion engendered by congruent sensory feedback.

In this paper, we advance a framework for the expressive sonification of a continuous ground contact, applicable for driving type experiences (e.g. biking). More specifically, we aim towards driving experiences, where a part of the VR interaction includes forward-moving wheels, making contact with surface material, while moving across the virtual landscape. The paper presents a preliminary case scenario using an instrumented training bicycle, for interacting with two surface textures: asphalt and dirt road (covered with leaves). Using image texture features and their intrinsic structure derived from displacement maps, we aim to foster a method to dynamically map repetitive user actions to a CSS engine. In this context, the main novelty is the use of CSS for aural and vibrotactile feedback, which to the best of our knowledge, has not been thoroughly explored and tested. Ultimately, the contribution aims to explore a larger framework that addresses spatiotemporal rolling friction with a high level of visual-aural-haptic synchronism and congruence. We show some promising preliminary results, which propose that sound descriptor-based models within CSS, can provide a real-time adaptive method to induce vibrotactile feedback within VR.

## 2. RELATED WORK

The human somatosensory (haptic) system is commonly divided into cutaneous and kinesthetic senses. Cutaneous, also frequently referred to as tactile sensation, refers to exteroceptors responding to stimuli across the entire surface of the skin, including touch, pressure, vibration, temperature and pain [7]. Kinesthetic sensation refers to the perceptual receptors in the joints, tendons and muscles which give information about position and movement of the body (proprioception). It constantly monitors if body movement is caused by self-directed motion or an external force (such as the vibration from the bike while crossing rough terrain) [8]. Haptic technology has the ability to evoke human somatosensory stimulation, by providing tactile or kinesthetic cues through e.g. ground-referenced haptic devices [9], for example, force feedback or vibration to physical steering props [10].

On a real bicycle ride, several sensory channels assist each other in maintaining balance and control, and vision is one of the most predominant modalities to elicit appropriate responses. However, other mechanical sensory systems are equally important to perform complex motor tasks. The vestibular system is responsible for sensing head movement, orientation and balance, and is tightly linked to the kinesthetic sensation and the visual system [7]. The multisensory integration of these different sensations assist humans in creating a unitary understanding of the world, e.g. through active haptic exploration. For example, the visual system can, with a greater range, anticipate incoming obstacles and changes in surface structure, which necessitates preparation of the body to respond accordingly, with both speed and precision to uphold balance and stability. Thus, recognition of surface textures and geometric varia-

tions is paramount to maintaining control of a real life bike. Equally, the perception of texture from a tactile perspective, can be viewed as a product of vibration during surface exploration with a lateral movement of the hand [11], or vibrotactile stimulation of the feet [3]. In VR, this sensation can be simulated through active haptics, such as vibrotactile actuators or force feedback systems [9]. It has been shown that mediating such information can increase the sense of realism of a virtual experience [12, 13], aid in the identification of surface information [14], as well as improving task performance and precision [15, 16].

Previous studies have shown that the human post-perceptual system will integrate conflicting information, in which case a bias towards the stronger modality may appear, also known as intersensory bias [17] or sensory dominance [2]. One of the most famous example of this, is the visuotactile cross-modal interaction of the rubber hand illusion [18]. In the experiment, a majority of subjects perceived a rubber hand as their own hand, when observing the rubber hand being brushed simultaneously as their own hand, while the real hand was in the same approximate position, but visually hidden. This sensory illusion persists, because the visual cues dominate the sense of proprioception. Furthermore, another study has shown that when exposed to a visuotactile discrepancy, the haptic sensation adapts to vision when visual stimuli is more reliable, but haptic exploration dominates vision when the reliability of the visual stimuli was decreased [19]. Similar results of haptic dominance was shown by [20] when participants were presented with ambiguous visual cues.

The perceptual system will go to great length to form and maintain a unitary experience. But while being an adept and capable system, it also shows a considerable reliance on specific pattern interpretations, and may easily be affected when its logic is challenged, whether intentionally or accidentally. Especially considering the latter; the range of multisensory feedback potential with immersive VR technology is comprehensive. The illusion of being present in the VE, requires that it represents a coherent perceptual experience with sensory consistency [21]. So while sensory integration from congruent stimuli may aid the intelligibility of an object or amplify the experience of its sensory feedback, incongruent stimuli may quickly become disruptive, confusing, or even hinder a sense of presence.

For VR experiences, where a central user experience (or interaction design aspects) depends on correct perceptual interpretations of VE content/objects (e.g. for interpretation of valid actions), consistent and congruent stimuli should be a crucial consideration [21]. Studies on natural interactive walking has shown how exactly auditory and active haptic feedback has been able to convey the multisensory experience of walking surfaces [3]. For an immersive experience of *driving* across simulated virtual surfaces, real-time auditory and vibrotactile feedback mediated with technology of high temporal and spatial resolution, is naturally likely to play an essential part as well, but differs from human plantigrade gait due to the mechanical differences between the interrupted surface contact of the



feet and that of a spinning wheel which maintains constant and continuous contact.

While studies on exercise biking show that virtual nature environments can be used to augment the exercise experience for motivational purposes with elderly users [22], these augmentations focus primarily on visuals for environment mediation. Experiencing nature content and traveling the environments, showed to be defining parts of elderly users' motivation to exercise [23]. For some users, introducing a VR headset to increase system immersion [21] further improved motivation and the user experience, partly due to experiencing of increased presence, but more qualitatively from feeling 'closer' to the (virtual) nature environment [24].

## 2.1 Visual world capture

For realistic experiences of feedback, e.g. from interactive virtual objects, congruence logically depends on the perceived realism of objects themselves, through all sensory modalities. In a study on the perceptual relationship between audio, video and audiovisual quality, results showed that high image quality positively affects the perception of (accompanying) audio quality, and vice versa [25, 26]. As the quality of the visual display is steadily increasing, it can be cogently derived that rendering techniques for interfaces targeting other sensory modalities must be paid equal attention and enhancement, to maintain sensory consistency. Photo-realistic capturing techniques, such as photogrammetry, can be supplied by affordable camera hardware (e.g., a standard smartphone-embedded camera) with tangible post processing time, to enable accessible 3D 'scanning' procedures of real-life objects. These captures can be merged into 3D geometry meshes, or surface capture for texture mapping, etc. [27]. The process includes combining a vast amount of photographs of an object from different angles for perfect alignment and capture of depth, to create a mesh and textures that resembles the captured object. High resolution meshes can be simplified in several ways, but often includes creating normal maps to simulate the fine geometric surface detail, and thus achieving the same level of detail and perceived granularity, with a decreased amount of vertices. However, the issue with normal maps is that the meshes themselves are often flat, which will be revealed when perceived from extreme angles. Displacement mapping is a different technique used to render details on a simple mesh, but where surfaces are actually displaced, based on a greyscale texture map (also known as a displacement map). Displacement maps are derived from height fields and elevation is encoded in each texel in a range from 0 (black) to 255 (white), in an 8-bit image, generating an actual displacement of the vertices along the surface normals [28]. Black-colored pixels are translated to minimum height, and white-colored pixels are translated to maximum height. Displacement maps typically requires a high level of vertices to achieve a good result, however some displacement procedures also afford a tessellation phase (i.e. adding additional subdivisions to the mesh before applying the displacement itself) [29].

Fig. 1 shows the process of adding geometry to a mesh

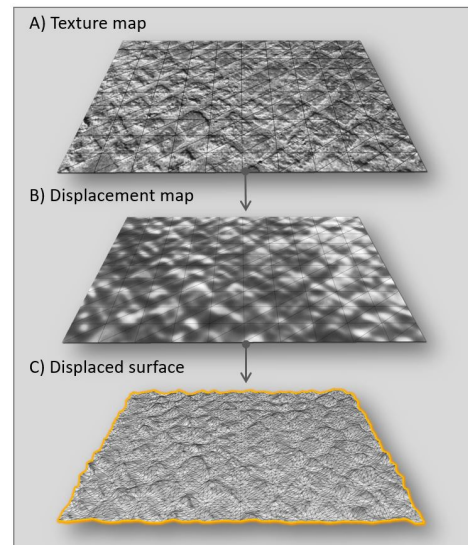


Figure 1. Texture mapping with an added displacement map. The results is a visually noticeable geometrical change of the surface, based fundamentally on the original texture map.

based on a displacement map and additional subdivision of the mesh (c). Here a texture map (a) is added to a primitive plane shape, and based on a displacement map (b) the mesh is tessellated and vertices are displaced along surface normals.

Although displacement mapping is a great way of achieving additional sense of depth and details, the addition of vertices also means an increased load on the graphical processing unit (GPU) [29]. For the developer, considerations towards optimization are always relevant, often related to the conflict of performance versus quality. For the purposes of this paper, the visual rendering of surfaces specifically (but not necessarily the remaining environment), should prioritize quality (displacement- over normal maps). Displacements and other geometric details in the visual surface representation, should be visually observable. Especially for moving forward, vision allows a user foresight of what surface features the vehicle will cross momentarily. It sets expectations for the translation of the visual cues into multimodal surface feedback. This can be the case for more prominent objects on the surface, but also simply for parts of the surface that are distinguishable or unique. And example would be a hard rock during on a soft dirt path, or a pile of crisp leaves on a sidewalk. The former (rock) would likely rely on displacement with multimodal feedback being focused quite a bit on the haptics, while the latter (leaves) would unlikely demand much displacement or haptic feedback, but would elicit strong auditory feedback.

## 2.2 Concatenative Sound Synthesis

Concatenative sound synthesis (CSS) is a sampling technique that creates audio streams by combining snippets from a large audio database [30]. It can be seen as an ex-

tension of granular synthesis [6,31], towards greater levels of control and automation by adopting content-based audio description methods from Music Information Retrieval (MIR).

Historically, CSS has been largely applied to the synthesis of sound textures/environmental sounds and music [32]. In the former category, of great relevance to the current work, CSS successfully tackles the pervasive post-production problem of extending a given audio clip, [33,34], to more creative solutions within games, VR and interactive installations; and even to procedurally generate highly controlled nuances that match external actuators [31,35].

CSS builds up a database of pre- or live-recorded *units* (i.e., segments or snippets with typical lengths of 50 ms to a few seconds) from an input *audio source*. Relevant sonic properties of each unit, such as pitch, loudness, noisiness, or spectral shape, are merged into a *feature vector*, which represents the units in the system. Due to their reduced dimensionality compared with the raw audio signal, feature vectors allow efficient search and retrieval from extensive data bases. New audio streams are created by specifying target queries, for which the best match is retrieved from the database to be played back.

The selection of attributes in the feature vector and target queries, as well as the metrics used to compare them are crucial to the system performance and quality. In this context, the nature of the input audio source, application domain and target definition (by navigating in a descriptor space) are fundamental to the parametrization of the system algorithms. For a comprehensive comparison of these variables and their implications in the musical results, please refer to [36].

### 3. METHODOLOGY

Fig. 2 shows the architecture of a concatenative sound synthesis engine for aural and vibrotactile feedback in VR. To the CSS prototypical component modules of this synthesis technique (in grey), we introduce a novel target definition method driven by texture map features from photogrammetric models of the provided surfaces (identified by their ID). Within this application context, particular emphasis is given to repetitive activities, such as walking, pedaling, and swimming, whose attributes are defined in the system by their angular velocity. Next, we detail each of the component modules of the architecture, using as a case-base scenario a bicycle ride on two surfaces asphalt and dirt road covered with leaves.

#### 3.1 Source sounds

We recorded real bicycle rides on two different surfaces: asphalt and dirt road covered with leaves. The choice of these surfaces aims at designing a preliminary battery of multimodal tests, which enforce scenarios where the vibrotactile and aural feedback is known to have different impact. While the vibrotactile feedback of these two surfaces was expected to have a minimal discrepancy, the aural feedback was expected to be quite distinctive. At two

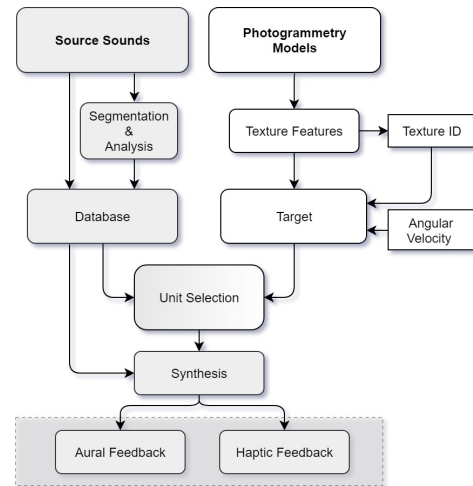


Figure 2. System architecture for a concatenative sound synthesis engine for vibrotactile and aural feedback (grey blocks) in VR. Target definition is driven by texture features from photogrammetric models, texture ID and angular velocity (white blocks).

of the recording sessions, microphones were mounted to a bicycle with 1) a *standard* cardioid microphone, pointing at the bicycle wheel to capture audio generated by the tire against the surfaces, and 2) a contact microphone attached to the chassis of the bike to capture the vibrations propagating through the wheel caused by the changing surface texture. The adopted audio capture techniques aimed at a clear distinction between synchronized auditory sources to test both the descriptors efficiency in discriminating the sources.

#### 3.2 Segmentation and analysis

To segment the recordings, we chopped the signal into equal-length units of 50 ms. This simple and efficient segmentation method was favored instead of more complex and structure-aware segmentation methods (e.g. using peaks from a spectral novelty function), as the source sounds are largely monotonic and repetitive in nature. Furthermore, the adopted unit length typically ensures a high degree of stationary behavior across its duration, thus promoting a more robust analysis of the signal.

Each unit was then analyzed using the entire set of eight low-level audio descriptors within MUBU [37]: frequency, energy, periodicity, first-order autocorrelation coefficient (AC1), loudness, spectral centroid, spectral spread, and spectral skewness. Hence, an eight-dimensional feature vector was created for each unit. Each descriptor per unit is represented by the mean value of overlapping windows across its duration (window size  $\approx 11.6ms$  with 50% overlap).

#### 3.3 Database

A hierarchical database architecture was adopted to store all imported source audio files and generated data. The former is stored into a buffer and can be easily retrieved by

accessing chunks of audio by an index (or sample) range. The latter stores all generated data from the source files during segmentation and analysis in a database using a hierarchical structure. Its top-level hierarchy includes as many entries as the number of surfaces, identified by an ID. Within each surface ID, we can parse a sub-level with three fields: 1) unit number, 2) unit onset (in samples), and 3) descriptors (feature vector).

### 3.4 Target

Targets are defined from displacement maps information per texture, which is acquired from the surface shaders in the Unity3D game engine. Acquiring surface information is a process that requires several steps. First of all, to measure the speed of the bike, the angular velocity has to be converted to meters per second, and the position of the bike in world space (i.e. the coordinate system of the game scene) has to be logged to figure out which mesh the bike is currently interacting with. In Unity, each game-object can be assigned a unique identification tag. Identification of surfaces is thus registered through a raycasting algorithm which originates from the approximated ground contact point of the front wheel of the virtual bike. When the raycast registers a collision with a ground-surface it identifies the surface-tag, and accesses the displacement map in the subshader, to read the corresponding normalized pixel value (between 0 and 1) in the texture coordinate (texcoord), defined by the position of the raycast-hit (see Fig.3.).

### 3.5 Unit selection

Units selection is the component module responsible for finding the best matching units to be synthesized. It aims to assess the database unit that best fits a particular target query—measured by its target cost, or distance in the descriptor space—but also by minimizing the spectral displacement between unit bounds—measured by concatenation cost. In this context, we focus mainly in the first metric, as the latter metric is implicitly modelled in the target definition. The better the targets capture the temporal nuances of the repetitive actions, the more unnecessary is the concatenation cost. Therefore, a simple real-time local search for the best matching unit is used without considering its surrounding temporal dimensions. Furthermore, we consider the adoption of jitter as a result of a flexible  $k$ -nearest neighbour with a user-definable  $k$  value, to be empirically tested.

### 3.6 Synthesis

Synthesis is done by two concatenating synthesis engines, which generate unique streams for each capture technique, i.e., aural and vibrotactile. These streams are then played back through their correspondent channel. The aural feedback is sent to the headphones and the vibrotactile feedback is sent to a low frequency audio transducer (Butt-Kicker BK-LFE).

Within each stream, selected units are concatenated with a Gaussian amplitude envelope and an 50% overlap. We adopt a spectral compressor-expander filter to enhance

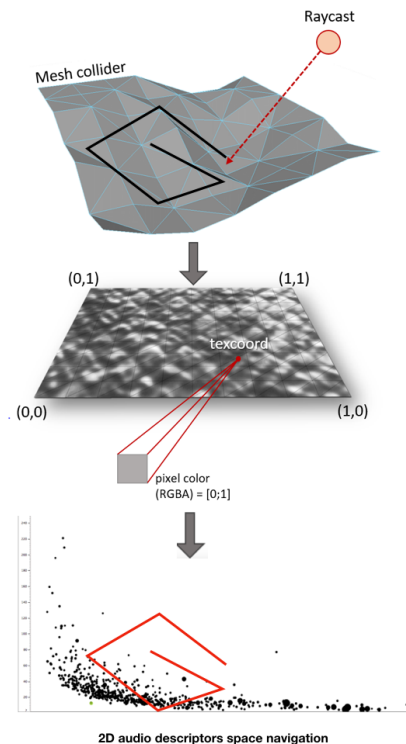


Figure 3. This figure visualizes the procedures of 1) acquiring surface information from a mesh through a raycast-hit, and 2) reading the corresponding texture-coordinate (texcoord) value in the displacement map, and 3) continuously sending that information to target units within the descriptors space once every frame.

the quality of the unit concatenation by minimizing discontinuities between the spectral peaks of adjacent units. Roughly, the signal process is done by applying on short-term windows ( $\approx 50\text{ms}$ ) a filtering mask resulting from the interpolation of the spectral content from neighbor windows.

## 4. PRELIMINARY EVALUATION

A preliminary evaluation of our framework aims to assess particular design choices of a unified CSS engine for both the aural and vibrotactile feedback in VR. In detail, we intend to promote a better understanding of the idiosyncrasies of the descriptor spaces resulting from our dual capture technique. Ultimately, we can endorse our informal perceptual hypothesis that the most expressive and salient features in aural and haptic descriptor spaces differ, and thus require different modelling strategies. To this end, we conducted statistical analysis to assess 1) which descriptors better represent each surface in a 2-D navigable space and 2) if the dual aural-haptic synthesis require different descriptors spaces to optimally navigate the corpus.

Following [34,38,39], we adopt a coefficient of variation,  $C_v$ , as a dimensionless (i.e. scale-invariant) measure of dispersion to identify the descriptors with greater salience.

As such, we aim to better discriminate the sound source in a 2-D navigable space and (conceivably) at a perceptual level. The coefficient of variation,  $C_v$ , is computed as the ratio of the standard deviation,  $\sigma$ , to the mean,  $\mu$ , so that:

$$C_v = \frac{\sigma}{\mu} \quad (1)$$

The result expresses a percentage value of the descriptor extent of variability in relation to its mean.

Moreover, we adopt the Spearman's rank correlation  $\rho$  to determine the statistical dependence between the contact and standard (condenser) microphones on their descriptors degree of variability (or saliency). The Spearman rank correlation is expressed by a value in the +1 to -1 range. +1 indicates a perfect association and -1 indicates a perfect negative association of ranks. The closer to 0, the weaker the association between the variables.

Our informal perceptual expectation is that the asphalt should provide a higher degree of descriptor dependency, given its low aural and haptic feedback, but may be more noticeable in surfaces such as the dirt road with leaves, as its aural feedback is notoriously more prominent than the haptic feedback.

Table 1 shows the coefficient of variation,  $C_v$ , and mean values,  $\mu$ , descriptor statistics for the two surfaces under study. The results are quite expressive across both surfaces and capture techniques for the spectral low-level timbral descriptors (centroid, spread, skewness and kurtosis). Conversely, the remaining temporal domain descriptors (frequency, energy, periodicity and first-order autocorrelation coefficient or AC1) show considerably less saliency. Moreover, the most salient descriptors (with the two highest coefficient of variation,  $C_v$ , for asphalt are the same for both capture techniques (contact and condenser), while for the dirt road with leaves surface, the descriptors correspondence does not hold. To a certain extent, this reinforces our perceptual hypothesis that both haptic and aural realities are different and require different corpus and navigation models. The Spearman's rank correlation for each surface across the two capture techniques shows a high degree of dependency between the contact and standard (condenser) microphones ( $\rho = .857$  with  $p > 0.01$ ) for the asphalt surface. A weaker association is found for the dirt road with leaves ( $\rho = .686$  with  $p > 0.05$ ). These results align with our initial expectations that some surfaces will demand a greater degree of separation in the modelling of trajectories and descriptor space definition (to optimize the 2-D navigation).

Fig. 4 presents the 2-D navigation plots per audio capture technique, in which both surfaces are included. The graphical representation suggests that some level of overlap between surfaces can be adopted in the architecture and system design. In other words, we can envision a possible scenario where all surfaces coexist in the same corpus, without the need to specify in the target a surface ID to constrain the unit search.

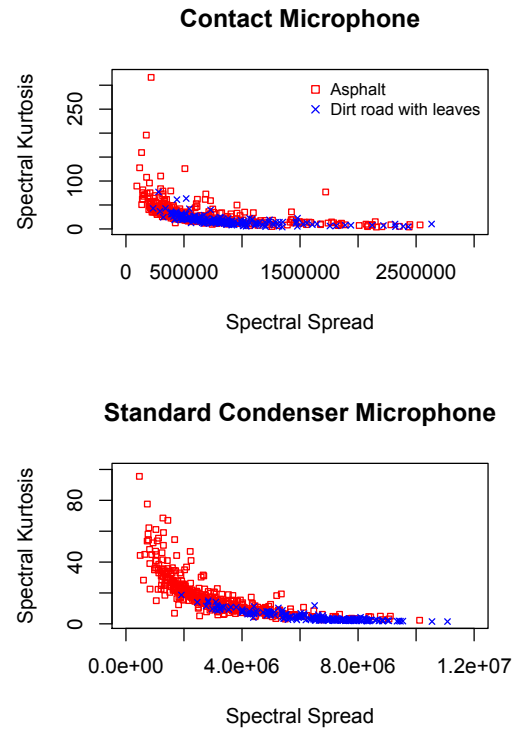


Figure 4. Two-dimensional visualization of the spectral spread and spectral kurtosis descriptor space for the two surfaces under study, i.e., asphalt and dirt road with leaves and for the two miking methods.

## 5. CONCLUSIONS AND FUTURE WORK

This paper presents preliminary work towards creating a novel approach to synthesize audiohaptic feedback from surface textures capture through e.g. photogrammetry. Besides facilitating visually realistic virtual objects, methods like photogrammetry represent new possibilities in a broader scheme. The ability to visually 'capture' reality into a realistic virtual representations, promotes methodological considerations for similar approaches to other modalities. The puzzle of constructing congruent integration of (synthetic) feedback, could become close to trivial, if feedback is constructed from capture of the actual visual, auditory and vibrotactile features of the real world object.

This work implementation and preliminary testing revealed that the combination of CSS with high realistic graphic assets, can provide a convincing and promising method to induce realism within immersive virtual environments. Furthermore, our preliminary results seem to indicate that using specific descriptors to analyze sound sources obtained from different capture methods can provide more precise clustering of sound units and sequences for each type of sensory feedback, including a further look into applications for realistic surface-haptic driving simulations.

We observed some limitations as well, at this point and related to some CSS intrinsic characteristics. A narrow



Audio Descriptor	Asphalt				Dirt with Leaves			
	Contact Microphone		Condenser Microphone		Contact Microphone		Condenser Microphone	
	$C_v$	$\mu$	$C_v$	$\mu$	$C_v$	$\mu$	$C_v$	$\mu$
Frequency	13	502.800	22	449.211	13	499.54	23	370.001
Energy	38	.005	32	.007	39	.009	34	.001
Periodicity	18	.371	41	.153	19	.388	53	.066
AC1	1	.967	1	.979	1	.963	2	.966
Loudness	7	-48.292	6	-47.194	7	-42.534	5	-40.083
Centroid	35	894.746	41	1294.465	30	1020.219	34	2846.533
Spread	<b>70</b>	692946.875	<b>57</b>	2942106.250	<b>50</b>	891232.511	27	6386084
Skewness	38	4.024	37	3.468	32	2.983	<b>72</b>	1.197
Kurtosis	<b>89</b>	31.758	<b>67</b>	20.014	<b>58</b>	17.701	<b>67</b>	4.681

Table 1. Coefficient of variation,  $C_v$ , and mean values,  $\mu$ , statistics for the asphalt and dirt road covered with leaves surfaces per audio descriptor and for the two audio capture techniques.

sound corpus, if sound database is not broad enough. And lacking audio representations of specific interactions might result in less expressive and inaccurate sensory feedback.

This paper marks the first in a ongoing line of studies using the model proposed. Studies of interest include approaches to practical implementations of the feedback system (especially haptics, most likely), users perception of realism based on the multimodal surface feedback, possible implications for vection with VR bike augmentation, presence studies with elderly users, and a line of perception tests to further explore the best practices of the multi-sensory balancing between the modalities and techniques. Furthermore, future research should at least consider the impact on system performance when introducing shaders that are more demanding to the GPU, such as displacement mapping. However, alternative methods exists that do make use of the information derived from height fields to simulate displacement without adding additional geometry, e.g. parallax mapping. Such alternatives could still utilize the same methodology while being more affordable in terms of computational resources. A fundamental topic in the near future is to test the use of accelerometers as capture technique of the surface displacements. Our aim is to learn which capture technique better drives the generation of vibrotactile feedback using CSS, towards a definite methodology and a fully working prototype.

## Acknowledgments

This work was supported by the Portuguese Foundation for Science and Technology (PD/BD/114140/2015).

## 6. REFERENCES

- [1] Y. Visell, F. Fontana, B. L. Giordano, R. Nordahl, S. Serafin, and R. Bresin, "Sound design and perception in walking interactions," *International Journal of Human-Computer Studies*, vol. 67, no. 11, pp. 947–959, 2009.
- [2] S. J. Lederman and R. L. Klatzky, "Multisensory texture perception," *The handbook of multisensory processes*, pp. 107–122, 2004.
- [3] S. Serafin, L. Turchet, R. Nordahl, S. Dimitrov, A. Berrezag, and V. Hayward, "Identification of virtual grounds using virtual reality haptic shoes and sound synthesis," in *Proceedings of eurohaptics symposium on haptic and audio-visual stimuli: enhancing experiences and interaction*, 2010, pp. 61–70.
- [4] R. L. Klatzky and S. J. Lederman, "Multisensory texture perception," in *Multisensory object perception in the primate brain*. Springer, 2010, pp. 211–230.
- [5] A. Di Scipio, "Synthesis of environmental sound textures by iterated nonlinear functions," in *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, 1999, pp. 109–117.
- [6] D. Schwarz, "Concatenative sound synthesis: The early years," *Journal of New Music Research*, vol. 35, no. 1, pp. 3–22, 2006.
- [7] S. M. Breedlove and N. V. Watson, *Behavioral neuroscience*. Sinauer Associates, Incorporated, Publishers, 2017.
- [8] M. S. Gazzaniga and T. F. Heatherton, *Psychological science: Mind, brain, and behavior*. WW Norton New York, 2003.
- [9] J. J. LaViola Jr, E. Kruijff, R. P. McMahan, D. Bowman, and I. P. Poupyrev, *3D user interfaces: theory and practice*. Addison-Wesley Professional, 2017.
- [10] D. C. Brogan, R. A. Metoyer, and J. K. Hodgins, "Dynamically simulated characters in virtual environments," *IEEE Computer Graphics and Applications*, vol. 18, no. 5, pp. 58–69, 1998.
- [11] S. J. Lederman and R. L. Klatzky, "Hand movements: A window into haptic object recognition," *Cognitive psychology*, vol. 19, no. 3, pp. 342–368, 1987.
- [12] L. Turchet, P. Burelli, and S. Serafin, "Haptic feedback for enhancing realism of walking simulations," *IEEE transactions on haptics*, vol. 6, no. 1, pp. 35–45, 2013.
- [13] S. Lind, L. Thomsen, M. Egeberg, N. Nilsson, R. Nordahl, and S. Serafin, "Effects of vibrotactile stimulation during virtual sandboarding," in *2016 IEEE Virtual Reality (VR)*. IEEE, 2016, pp. 219–220.

- [14] M. R. McGee, P. Gray, and S. Brewster, "Haptic perception of virtual roughness," in *CHI'01 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2001, pp. 155–156.
- [15] I. Oakley, M. R. McGee, S. Brewster, and P. Gray, "Putting the feel in 'look and feel'," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 2000, pp. 415–422.
- [16] V. Nitsch and B. Färber, "A meta-analysis of the effects of haptic interfaces on task performance with teleoperation systems," *IEEE transactions on haptics*, vol. 6, no. 4, pp. 387–398, 2013.
- [17] R. B. Welch and D. H. Warren, "Immediate perceptual response to intersensory discrepancy," *Psychological bulletin*, vol. 88, no. 3, p. 638, 1980.
- [18] M. Botvinick and J. Cohen, "Rubber hands 'feel' touch that eyes see," *Nature*, vol. 391, no. 6669, p. 756, 1998.
- [19] J. Burge, A. R. Girshick, and M. S. Banks, "Visual-haptic adaptation is determined by relative reliability," *Journal of Neuroscience*, vol. 30, no. 22, pp. 7714–7721, 2010.
- [20] M. O. Ernst, M. S. Banks, and H. H. Bühlhoff, "Touch can change visual slant perception," *Nature neuroscience*, vol. 3, no. 1, p. 69, 2000.
- [21] M. Slater, "Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 364, no. 1535, pp. 3549–3557, 2009.
- [22] J. R. Bruun-Pedersen, S. Serafin, and L. B. Kofoed, "Restorative virtual environment design for augmenting nursing home rehabilitation," *Journal For Virtual Worlds Research*, vol. 9, no. 3, 2016.
- [23] —, "Motivating elderly to exercise-recreational virtual environment for indoor biking," in *Serious Games and Applications for Health (SeGAH), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–9.
- [24] —, "Going outside while staying inside—exercise motivation with immersive vs. non-immersive recreational virtual environment augmentation for older adult nursing home residents," in *Healthcare Informatics (ICHI), 2016 IEEE International Conference on*. IEEE, 2016, pp. 216–226.
- [25] A. Kohlrausch and S. van de Par, "Auditory-visual interaction: from fundamental research in cognitive psychology to (possible) applications," in *Human Vision and Electronic Imaging IV*, vol. 3644. International Society for Optics and Photonics, 1999, pp. 34–45.
- [26] J. G. Beerends and F. E. De Caluwe, "The influence of video quality on perceived audio quality and vice versa," *Journal of the Audio Engineering Society*, vol. 47, no. 5, pp. 355–362, 1999.
- [27] S. Lachambre, S. Lagarde, and C. Jover, "Unity photogrammetry workflow," 2017. [Online]. Available: [https://unity3d.com/files/solutions/photogrammetry/Unity-Photogrammetry-Workflow\\_2017-07\\_v2.pdf](https://unity3d.com/files/solutions/photogrammetry/Unity-Photogrammetry-Workflow_2017-07_v2.pdf)
- [28] R. Fernando and M. J. Kilgard, *The Cg Tutorial: The definitive guide to programmable real-time graphics*. Addison-Wesley Longman Publishing Co., Inc., 2003.
- [29] J. Birn, *Digital lighting & rendering*. Pearson Education, 2014.
- [30] M. Casey, "Soundspotting: A new kind of process?" in *The Oxford Handbook of Computer Music*, 2009.
- [31] D. Schwarz and N. Schnell, "Descriptor-based sound texture sampling," in *Proceedings of the Sound and Music Computing Conference*, 2010, pp. 510–515.
- [32] G. Bernardes and D. Cocharro, *Dynamic Music Generation, Audio Analysis-Synthesis Methods*. Cham: Springer International Publishing, In Press.
- [33] M. Frojd and A. Horner, "Fast sound texture synthesis using overlap-add," in *International Computer Music Conference*, 2007.
- [34] G. Bernardes, L. Aly, and M. E. Davies, "Seed: Resynthesizing environmental sounds from examples," in *Proceedings of the Sound and Music Computing Conference*, 2016.
- [35] G. Bernardes, C. Guedes, and B. Pennycook, *Ear-Gram: An Application for Interactive Exploration of Concatenative Sound Synthesis in Pure Data*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 110–129.
- [36] N. M. Norowi, E. R. Miranda, and M. Hussin, "Parametric factors affecting concatenative sound synthesis," *Advanced Science Letters*, vol. 23, no. 6, pp. 5496–5500, 2017.
- [37] N. Schnell, A. Röbel, D. Schwarz, G. Peeters, R. Borghesi *et al.*, "Mubu and friends—assembling tools for content based real-time interactive audio processing in max/msp," in *Proceedings of the International Computer Music Conference*, 2009.
- [38] W. Brent, "A timbre analysis and classification toolkit for pure data," in *Proceedings of the International Ccomputer Music Conference*, 2010.
- [39] L. Aly, R. Penha, and G. Bernardes, "Digit: A digital foley system to generate footstep sounds," in *Music Technology with Swing*, M. Aramaki, M. E. P. Davies, R. Kronland-Martinet, and S. Ystad, Eds. Cham: Springer International Publishing, 2018, pp. 429–441.

# Percussion Synthesis using Loopback Frequency Modulation Oscillators

Jennifer S. Hsu, Tamara Smyth

University of California San Diego, Department of Music

jsh008@ucsd.edu, trsmyth@ucsd.edu

## ABSTRACT

In this work, we apply recent research results in loopback frequency modulation (FM) to real-time parametric synthesis of percussion sounds. Loopback FM is a variant of FM synthesis whereby the carrier oscillator “loops back” to serve as a modulator of its own frequency. Like FM, more spectral components emerge, but further, when the loopback coefficient is made time varying, frequency trajectories that resemble the nonlinearities heard in acoustic percussion instruments appear. Here, loopback FM is used to parametrically synthesize this effect in struck percussion instruments, known to exhibit frequency sweeps (among other nonlinear characteristics) due to modal coupling. While many percussion synthesis models incorporate such nonlinear effects while aiming for acoustic accuracy, computational efficiency is often sacrificed, prohibiting real-time use. This work seeks to develop a real-time percussion synthesis model that creates a variety of novel sounds and captures the sonic qualities of nonlinear percussion instruments. A linear, modal synthesis percussion model is modified to use loopback FM oscillators, which allows the model to create rich and abstract percussive hits in real-time. Musically intuitive parameters for the percussion model are emphasized resulting in a usable percussion sound synthesizer.

## 1. INTRODUCTION

Synthesis of plates, membranes, and other percussion instruments have been realized using several different modeling techniques including modal synthesis (MS) [1], the Functional Transformation Method (FTM) [2], finite difference schemes (FDS) [3], and the digital waveguide mesh (DWM) [4]. When real percussion instruments are struck with a large velocity excitation, nonlinear effects often result. Examples of these nonlinearities include the cascade of energy from low to high frequency components that give cymbals their characteristic sound and pitch glides heard with gongs [5].

Many nonlinear percussion synthesis models are computationally expensive and may exhibit stability issues that render them unsuitable for real-time synthesis on standard computers [1–3]. An exception to this is described in [6], where computationally heavy calculations are approximated so that a nonlinear membrane model is able to

Copyright: © 2019 Jennifer S. Hsu et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

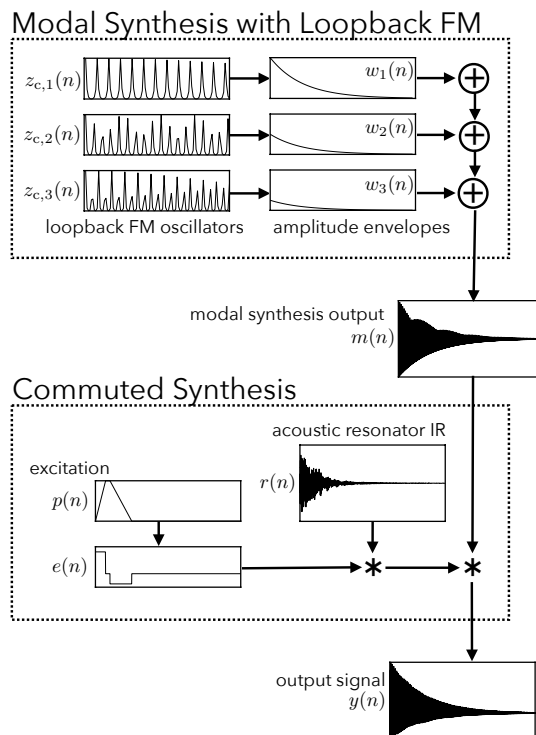


Figure 1. The loopback FM percussion synthesis method.

simulate up to 1000 modes in real-time at a sampling rate of 44.1kHz. For the percussion models in [7] and [2], the most computationally expensive component is the nonlinear calculation. Though the linear version of each model can be efficiently computed, they produce sounds that are less interesting than when nonlinearities are added. With this understanding, one may like to use a linear system to synthesize percussive sounds and approximate the nonlinearities in an efficient and perceptually similar way.

For example, in [2], Avanzini and Marogna present a sound synthesis simulation of a nonlinear, tension-modulated percussion membrane. The model consists of a linear portion and a nonlinear feedback section simulating tension modulation. The linear model is computationally efficient, but the nonlinear tension modulation requires a feedback calculation for every sample, a computational complexity that makes it unable to run in real time. In a following paper [8], the nonlinear feedback calculation is replaced with an efficient approximation that can be calculated with the computational expense similar to that of the linear model.

Instead of striving for an acoustically accurate simulation as some previous research has done, the aim here is to create a percussion synthesizer that creates a variety of novel sounds inspired by the dynamic and nonlinear phenomena heard in percussion instruments. Similar to the strategy used in [9], in which the pitch glide capabilities of a Duffing oscillator are explored in the sound synthesis of a gong, the work presented herein employs the nonlinear effects of loopback FM, a technique initially presented in [10] and further developed in [11]. Here, loopback FM oscillators are used to enhance a modal synthesis (MS) of percussion sound (see Figure 1). Loopback FM is a variant of FM synthesis where the carrier signal is looped back to modulate its own frequency, resulting in complex spectra (much like traditional FM), and interesting frequency trajectories that resemble the nonlinearities observed in real percussion instruments when the loopback coefficient is made time varying.

In Section 2, we explore traditional MS and how it can be used to synthesize percussive sounds. Section 3 reviews loopback FM equations relevant to the current context. Section 4 explains how traditional MS can be modified with loopback FM oscillators to create a wide variety of percussion sounds. Synthesis parameters are discussed in Section 5. Section 6 presents synthesis examples of a marimba, tom tom, and circular plate. Concluding thoughts and future research directions are considered in Section 7.

## 2. PERCUSSION SYNTHESIS USING TRADITIONAL MODAL SYNTHESIS

MS is a technique that resynthesizes the sound of an acoustic object according to its acoustic modes or vibrational patterns. The resonant frequencies of an acoustic object arise through the sinusoidal motion of the object's modes. With traditional MS, each mode is synthesized using a second-order resonating filter with a corresponding frequency, initial amplitude, and decay [12].

To synthesize a percussive sound with MS, we begin with a list of  $N_f$  modal frequencies  $f_i$ , the values of which can be obtained from acoustic experiments, spectral analysis of recorded or physically modeled sounds (e.g. DWMs, FDSs, etc), or calculated using theoretical equations.

Though MS traditionally models each frequency mode with a second-order bandpass resonant filter with center frequency  $f_i$ , in this work the filters are replaced by sinusoidal oscillators of frequency (or center frequency if frequency is time varying)  $f_i$ . This allows for a straightforward comparison with the loopback FM version (also implemented here with oscillators) than if traditional MS bandpass filters had been used. A sinusoidal component with carrier frequency  $\omega_{c,i} = 2\pi f_i$  is expressed as  $s_i(n) = \sin(\omega_{c,i}nT)$  for time sample  $n$  and period  $T = 1/f_s$  for sampling rate  $f_s$ . Each sinusoidal component is multiplied with an amplitude envelope  $w_i(n)$ . For percussive sounds,  $w_i(n)$  are typically exponentially decreasing envelopes with possibly different initial amplitudes and decay rates for different modes. For example, for natural sounding results, higher-frequency modes should be made to decay more rapidly. Enveloped sinusoidal components

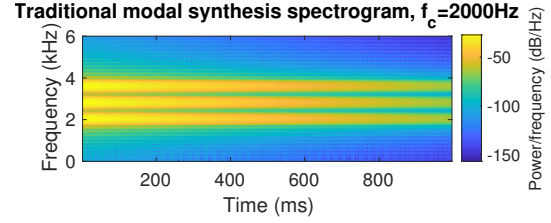


Figure 2. Traditional MS for three modal frequencies.

are added together to form the MS output given by

$$m_s(n) = \sum_{i=0}^{N_f-1} w_i(n)s_i(n), \quad (1)$$

the spectrum of which is shown in Figure 2 for  $N_f = 3$ .

MS is efficient and useful for recreating the sound of objects that consist of a small number of resonant frequency modes. However, MS is a linear method and (1) is incapable of capturing nonlinear effects. A simple modification to the sinusoidal components of the MS framework allows the system to create complex and dynamic sounds reminiscent of nonlinear vibrations in percussion instruments. In this modification, each sinusoidal component is looped back to modulate its own carrier frequency, a synthesis technique coined by the authors as “Loopback FM.”

## 3. LOOPBACK FM

Loopback FM is a self-modulated form of FM where the oscillator loops back and modulates its own carrier frequency according to a feedback coefficient. This differs from Feedback FM [13], in which the output is used to modulate its own initial phase. Loopback FM with a static feedback coefficient,  $B$ , and feedback FM both create peaks in the spectrum at integer multiples of a sounding frequency. As described in [10], the difference between the two synthesis methods is that with loopback FM, the feedback coefficient  $B$  can be varied over time to create both predictable pitch and spectral changes. Conversely, feedback FM preserves pitch (in some contexts a desirable feature) and only introduces spectral changes. As shown in [11], loopback FM and its closed-form IIR approximation, an expression that resembles the transfer function of a “stretched” allpass filter [14] but for which only the real part is used as a time-domain signal, can be used to create complex frequency spectra and pitch contours. Here, we present the equations for loopback FM and its closed form representation with static pitch and timbre followed by their time-varying formulations, which can be used to modulate timbre and sounding frequency.

### 3.1 Loopback FM Formulation

The loopback FM equation involves a carrier frequency  $\omega_c = 2\pi f_c$  where  $\omega_c$  is the angular frequency and  $f_c$  is the frequency in Hz, and a feedback parameter  $B$ , which controls the output's timbre and fundamental frequency.

The loopback FM equation for static  $B$  and time sample  $n$  is

$$z_c(n) = e^{j\omega_c T(1+B\Re\{z_c(n-1)\})} z_c(n-1), \quad (2)$$



with the initial condition  $z_c(0) = 1$  causing oscillation. The output that we listen to is the real part of  $z_c(n)$ . The fundamental frequency of this oscillator is not  $\omega_c$  but rather  $\omega_0 = 2\pi f_0$ , where  $f_0$  is the sounding frequency in Hz. The relationship between  $\omega_0$  and  $\omega_c$  is described by

$$\omega_0 = \omega_c \sqrt{1 - B^2}, \quad -1 \leq B \leq 1, \quad (3)$$

which shows that for  $\omega_0$  to remain real, the value of  $B$  must be within the interval  $(-1, 1)$ .

### 3.2 An Alternate Representation of the Loopback FM Oscillator

The loopback FM oscillator  $z_c(n)$  given in (2) with static pitch and timbre may also be represented by the closed-form representation

$$z_0(n) = \frac{b_0 + e^{j\omega_0 n T}}{1 + b_0 e^{j\omega_0 n T}}, \quad (4)$$

which is similar to the transfer function of a “stretched” all-pass filter used in [14]. In this synthesis context, (4) is used as a time-domain signal that is a function of time sample  $n$ , where  $b_0$  influences spectrum,  $\omega_0$  specifies the sounding frequency, and the sound is the real signal given by  $\Re\{z_0(n)\}$ . Parameters  $b_0$  and  $\omega_0$  are related to the loopback FM feedback coefficient  $B$ . With (4), timbre and pitch can be independently controlled, but this is not possible with loopback FM parameters  $\omega_c$  and  $B$  given in (2). Coefficient  $b_0$  in  $z_0(n)$  is related to loopback FM parameter  $B$  through

$$b_0 = \frac{\sqrt{1 - B^2} - 1}{B}. \quad (5)$$

Note the singularity in (5) for  $B = 0$ . The relationship between  $\omega_0$  and  $\omega_c$  is shown in (3).

### 3.3 Time-varying $B$ : Pitch and Timbre Modulation with $z_c(n)$

In (2), the feedback coefficient  $B$  can be varied over time between  $(-1, 1)$  to create pitch glides and timbre variations over the length of the output signal. From (2),  $B$  is replaced by  $B(n)$  to form

$$z_c(n) = e^{j\omega_c T(1+B(n)\Re\{z_c(n-1)\})} z_c(n-1) \quad (6)$$

(3) reveals that when  $B$  is made to vary over time,  $\omega_0$  also becomes time-varying. This creates a pitch trajectory where the sounding frequency follows

$$\omega_0(n) = \omega_c \sqrt{1 - B^2(n)} \quad (7)$$

### 3.4 Time-varying $b_0$ and $\omega_0$ : Pitch and Timbre Modulation with $z_0(n)$

Like (6), the parameters of  $H$  in (4) can be made to vary over time to create pitch glides and spectral changes. Parameter  $b_0$  can be mapped to  $B(n)$  by

$$b_0(n) = \frac{\sqrt{1 - B^2(n)} - 1}{B(n)} \quad (8)$$

and used in (9) to create spectral variations.

A desired pitch contour can be created by setting  $\omega_0(n)$  to a pitch trajectory in the form of (7). Directly using  $\omega_0(n)$  in place of  $\omega_0$  in (4) will not result in the desired pitch glide, and it is necessary to use a generalization of (4):

$$z_0(n) = \frac{b_0(n) + e^{j\Theta_0(n)}}{1 + b_0(n)e^{j\Theta_0(n)}}. \quad (9)$$

To understand  $\Theta_0(n)$ , the instantaneous phase of the complex exponential terms in (9), let  $\omega_0(t)$  be the continuous counterpart of  $\omega_0(n)$  serving as the instantaneous frequency, and  $\Theta_0(t)$  its integral with respect to time:

$$\Theta_0(t) = \int_0^t \omega_0(t) dt. \quad (10)$$

Examples of the discrete-time form of (10) given by  $\Theta_0(n)$  as used in (9), are shown in Section 5.5.

## 4. PERCUSSION SYNTHESIS WITH LOOPBACK FM OSCILLATORS

The main steps involved in the loopback FM percussion synthesis method are shown in Figure 1. The “Modal Synthesis with Loopback FM” block consists of synthesizing the vibrations of an abstract, nonlinear surface using MS and (6) or (9) to produce output  $m(n)$ . The “Commutated Synthesis” block completes the percussion model by convolving a parametric excitation function and acoustic resonator impulse response with  $m(n)$ .

### 4.1 Modal Synthesis with Loopback FM

Like the percussion MS technique described in Section 2, the “Modal Synthesis with Loopback FM” block begins with a list of modal frequencies  $f_i$  of length  $N_f$ . Instead of sinusoidal oscillators,  $N_f$  loopback FM oscillators are generated using the frequencies in  $f_i$ . As described in [11], the loopback FM oscillators can be expressed as resonating filters, though here, they are implemented as oscillators. This is similar to implementing MS with sinusoidal oscillators as opposed to resonating filters as described in Section 2. The loopback FM oscillator  $z_{c,i}(n)$  has been synthesized with carrier frequency  $\omega_{c,i} = 2\pi f_i$ , where subscript  $i$  means that  $\omega_{c,i}$  is set using the  $i^{\text{th}}$  frequency in  $f_i$ .

The real part of each loopback FM oscillator is multiplied with an amplitude envelope  $w_i(n)$  and the enveloped loopback FM oscillators are summed to create the MS output

$$m(n) = \sum_{i=0}^{N_f-1} w_i(n) \Re\{z_{c,i}(n)\}. \quad (11)$$

### 4.2 Commuted Synthesis

In [15], Smith efficiently models stringed musical instruments using commuted synthesis. This technique is adapted here for percussion synthesis.

To complete the percussion instrument model,  $m(n)$  must be excited by an excitation function,  $e(n)$ , and coupled to an acoustic resonator with impulse response  $r(n)$ . The equation to synthesize this relationship is

$$y(n) = e(n) * m(n) * r(n) \quad (12)$$

where  $*$  indicates convolution. Because there is no dependence between  $m(n)$ ,  $e(n)$ , and  $r(n)$ ,  $m(n)$  can be commuted with  $r(n)$ . The excitation and resonator impulse response can be convolved to form an aggregate excitation  $a(n) = e(n) * r(n)$ .

Aggregate excitations can be stored for several excitation and resonator combinations. During run-time, a low-latency convolution method, such as the one described in [16], can be used to convolve  $a(n)$  with  $m(n)$  to form the final percussion model output

$$y(n) = a(n) * m(n). \quad (13)$$

In our syntheses, we use a variety of resonator impulse responses as presented in Section 6 along with two different types of parametric excitations.

### 4.3 Excitations

The “Excitation” block in Figure 1 involves  $p(n)$ , a function that describes the vertical position of a drumstick/mallet hitting a surface at time  $n$ . The excitation signal  $e(n) = p(n) - p(n-1)$ , relates to the velocity of the drumstick/mallet and is convolved with the acoustic resonator impulse response to form  $a(n)$ . Here, we use raised cosine envelopes and filtered noise bursts for  $p(n)$ . These signals are parametric and affect the resulting output timbre.

#### 4.3.1 Raised Cosine Envelopes

The raised cosine envelope has a single parameter: the window length  $L$ . The equation for the excitation is

$$p(n) = \begin{cases} 0.5 \left( 1 - \cos \left( \frac{2\pi n}{L-1} \right) \right), & \text{for } 0 \leq n < L \\ 0, & \text{for } n \geq L \end{cases} \quad (14)$$

#### 4.3.2 Filtered Noise Bursts

The parameters for a filtered noise burst are noise burst duration  $t_d$  and low and high frequency cutoffs for a bandpass filter  $f_{\text{low}}$  and  $f_{\text{high}}$ . Examples in this paper use white noise filtered by a second-order Butterworth bandpass filter.

## 5. MUSICAL PARAMETERS FOR LOOPBACK FM PERCUSSION SYNTHESIS

Musical parameters for the loopback FM percussion synthesis method are presented here along with their corresponding variables and equations.

### 5.1 Timbre: Oscillators created with $z_{c,i}(n)$ or $z_{0,i}(n)$

The MS oscillators can be synthesized using  $z_{c,i}(n)$  or  $z_{0,i}(n)$ . To use  $z_{0,i}(n)$  oscillators, replace  $z_{c,i}(n)$  in (11) with  $z_{0,i}(n)$ . While both forms produce almost identical results from  $f_c = 0$  Hz to around  $f_c = 2500$  Hz at a sampling rate of 44.1 kHz, when  $f_c > 2500$  Hz, the version that uses  $z_{c,i}(n)$  becomes much noisier, due to aliasing. If the sampling rate is increased, the output is the same whether the oscillators are created using  $z_{c,i}(n)$  or  $z_{0,i}(n)$ . Like FM, both  $z_{c,i}(n)$  and  $z_{0,i}(n)$  produce signals

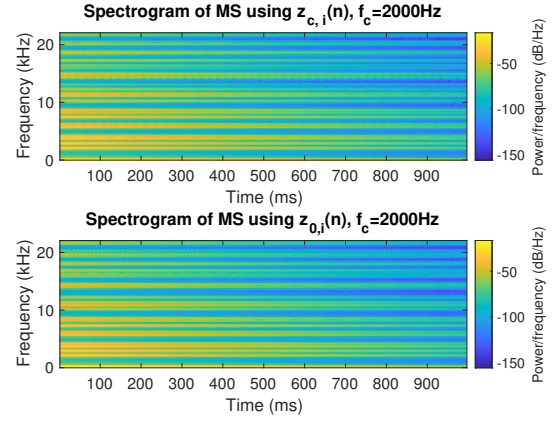


Figure 3. MS using  $z_{c,i}(n)$  and  $z_{0,i}(n)$  with low carrier frequencies create almost identical results.

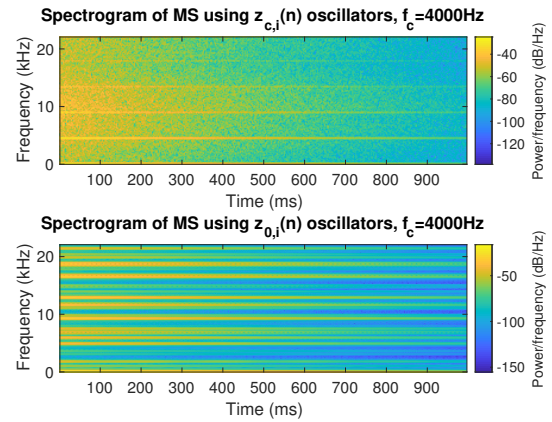


Figure 4. MS with  $z_{c,i}(n)$  creates noisier output than MS with  $z_{0,i}(n)$  for high carrier frequencies.

that are not bandlimited. With loopback FM, a large  $f_c$  means a large feedback amount, which can mean increased bandwidth and aliasing, similar to how a large index of modulation corresponds to a wider bandwidth in traditional FM. Figure 3 shows that the  $z_{c,i}(n)$  and  $z_{0,i}(n)$  MS oscillators produce similar spectrograms when the lowest of 3 modal frequencies is set to a low carrier frequency of  $f_c = 2000$  Hz. Vastly different spectrograms are produced when the lowest of the 3 modal frequencies is set to a higher carrier frequency of  $f_c = 4000$  Hz as shown in Figure 4. The MS using  $z_{c,i}(n)$  synthesizes a noisier output and can be used to create cymbal- and crash-like sounds as shown in Section 6.3.

### 5.2 Timbre: $B$ and $b_0$

Loopback FM parameter  $B$  controls timbre in (2) while  $z_0(n)$  parameter  $b_0$  affects timbre in (4). For carrier frequencies below 2500 Hz, the frequency components created using (2) or (4) are almost identical and are spaced at integer multiples of  $f_0$ . When  $B = 0$  or  $b_0 = 0$ , there are no sidebands and the output is a pure tone. As  $B$  and  $b_0$  increase towards 1 (or decrease towards  $-1$ ), more sidebands appear and the timbre brightens. The sidebands logarithmically decrease in amplitude for each multiple of  $f_0$ .

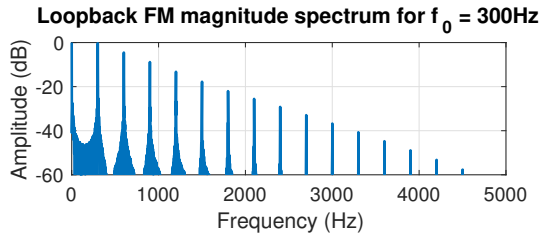


Figure 5. The loopback FM magnitude spectrum. Frequency components occur at integer multiples of the sounding frequency, 300Hz, and the amplitude of the components decreases logarithmically.

Figure 5 is a plot of the magnitude spectrum for a loopback FM oscillator with static  $B = 0.9$  and  $f_0 = 300\text{Hz}$ . The sounding frequency can be seen as a peak at 300Hz and the sidebands are spaced at integer multiples of 300Hz with a logarithmic decrease.

(5) explains the relationship between  $b_0$  and  $B$ , though as described in Section 5.1, at high carrier frequencies, the output from (2) will differ from that of (4).

### 5.3 Time-varying Timbre: $B(n)$ and $b_0(n)$

With (6),  $B(n)$  affects the time-varying timbre and sounding frequency. When using (9),  $b_0(n)$  controls the time-varying timbre, independent of pitch. As in the static case, as  $B(n)$  and  $b_0(n)$  near 0, the output approaches a pure tone, while as  $B(n)$  and  $b_0(n)$  approach 1 and  $-1$ , the number of sidebands created by the oscillators increases and the timbre becomes brighter.

In Figure 6,  $B(n) = g^n$  where  $g = 0.9999$ ,  $b_0(n)$  is obtained according to (8), and amplitude envelopes are the same for all modal frequencies. The top and middle plots in Figure 6 compare spectrograms for a static timbre of  $b_0 = -0.6312$  with (4) and a time-varying timbre where  $b_0(n)$  is used with (9). The sidebands in the top plot are the same over the course of the signal, but the higher frequency sidebands die out over time in the middle plot as  $b_0(n)$  increases from  $-1$  to 0. Time-varying timbre between (6) and (9) can be compared using the middle and bottom plots. In the bottom plot, time-varying  $B(n)$  creates timbre and pitch variation as  $n$  increases. In the middle plot,  $b_0(n)$  changes the timbre without affecting the frequency trajectories.

### 5.4 Sounding Frequency: $\omega_0$

For Eqs. 2 and 4, the sounding frequency can be controlled with  $\omega_0 = 2\pi f_0$ . For a desired  $\omega_0$  with (2), one would use (3) and either 1) set  $B$  to a desired value and solve for  $\omega_c$  or 2) set  $\omega_c$  and solve for  $B$ .

Because the modal frequencies for percussive instruments are often inharmonic, the sounding frequency for percussion synthesis is not clearly defined. With MS using  $z_{0,i}(n)$  oscillators,  $\omega_{0,i} = 2\pi f_i$  is used to set the sounding frequencies of individual oscillators. For MS using  $z_{c,i}(n)$  oscillators, the carrier frequencies can be set to the modal frequencies:  $\omega_{c,i} = 2\pi f_i$  or the sounding frequencies can be set to the modal frequencies:  $\omega_{0,i} = 2\pi f_i$ . According

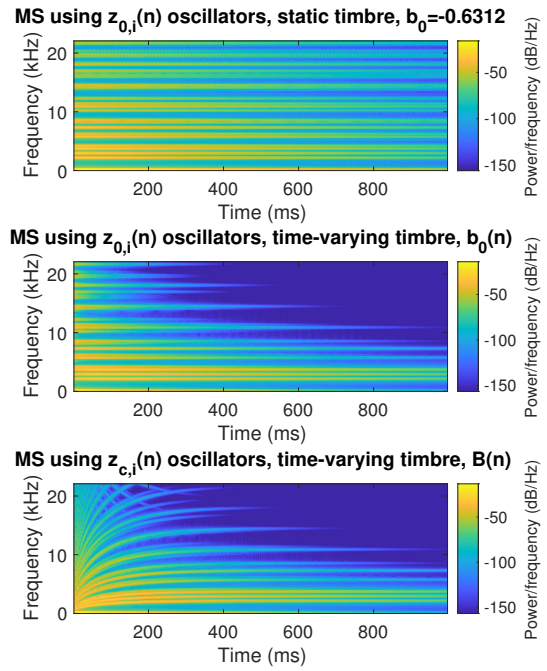


Figure 6. Static and time-varying timbre with various MS oscillators. Middle:  $b_0(n)$  with (9) modulates timbre independently of pitch. Bottom:  $B(n)$  with (6) affects both timbre and pitch.

to (3), when  $B = 0$ ,  $\omega_0 = \omega_c$ , and setting either to the modal frequencies would create the same output. When  $B$  is large and close to 1 or  $-1$ ,  $\omega_0$  will be a lower frequency than  $\omega_c$ . This means that using  $\omega_{c,i} = 2\pi f_i$  produces lower sounding frequencies while setting  $\omega_{0,i} = 2\pi f_i$  produces higher sounding frequencies, which will most likely produce aliasing effects, especially with (2), as described in Section 5.1. Figure 7 demonstrates that when  $B$  is close to 1,  $\omega_{c,i} = 2\pi f_i$  creates a toned output while  $\omega_{0,i} = 2\pi f_i$  creates a noisy output as higher frequencies contribute to extreme aliasing effects.

### 5.5 Pitch Glides: $B(n)$ and $\omega_0(n)$

With (6), a pitch glide can be added by varying  $B(n)$  over time. This also produces timbral changes. A pitch glide can be created with (9) by varying  $\omega_0(n)$  over time as described in Section 3.4. To modify the pitch independently of timbre with (9),  $b_0$  should be held constant.

As described in Section 5.1, differences between (6) and (9) can be observed when  $\omega_c$  is high, and this effect occurs with pitch glides. Figure 8 shows the high carrier frequency difference for a pitch glide over three modal frequencies using MS with (6) and (9). The pitch glide is created with  $B(n) = 0.9999^n$ , so timbre also changes. At higher carrier frequencies, MS with Eq. 6 creates noise-like output for the first 100ms and more spectral components than MS with Eq. 9 from 100 – 250ms. From 300ms through the remainder of the signal, the frequency components are more similar.

Pitch glides are constrained to use exponential and linear  $B(n)$  functions in the synthesis examples presented in this



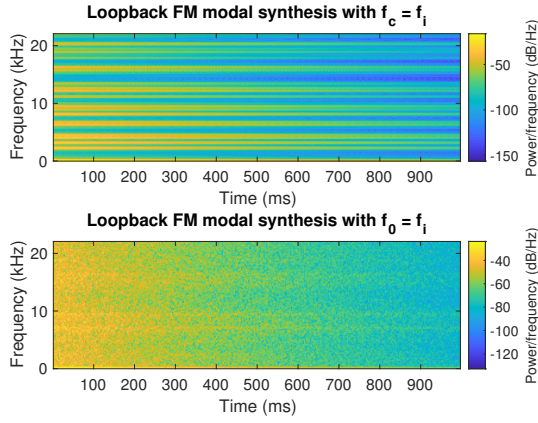


Figure 7. Loopback FM MS spectrograms for  $\omega_{c,i} = 2\pi f_i$  (top) and  $\omega_{0,i} = 2\pi f_i$  (bottom). The top and bottom signals are generated using the same 3 modal frequencies with  $B = 0.9$ .

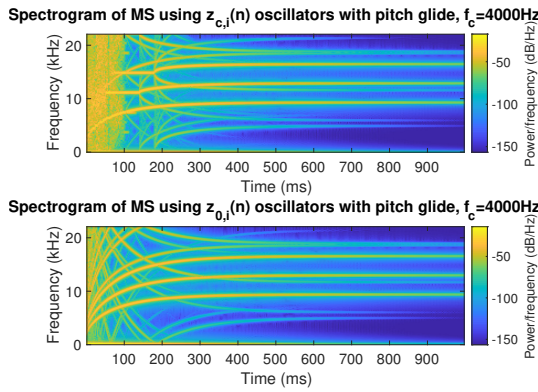


Figure 8. MS using (6) vs. (9) at high carrier frequencies with a pitch glide produce different spectrograms for the first 250ms. The lowest oscillator frequency uses  $f_c = 4000\text{Hz}$ .

research. This restriction on  $B(n)$  creates natural sounding pitch glides and allows us to draw parallels between the  $z_c(n)$  and  $z_0(n)$  forms.

### 5.5.1 Pitch Glides with Exponential $B(n)$

For the exponential case,  $B(n) = g^n$  can be used directly in (6) to produce a pitch glide. When using (9) for a pitch glide,  $\Theta_0(n)$  can be found using  $B(n) = g^n$  along with Equations 7 and 10 which, as shown in [11], is given by

$$\Theta_0(n) = \frac{\omega_c}{\log(g)} (\sqrt{1 - g^{2n}} - \tanh^{-1}(\sqrt{1 - g^{2n}})) + C \quad (15)$$

where  $C$  is the constant of integration.

### 5.5.2 Pitch Glides with Linear $B(n)$

For the linear case,  $B(n) = kn + l$  produces a pitch glide when used with (6). (9) uses the instantaneous phase given by

$$\Theta_0(n) = \frac{\omega_c}{2k} ((kn + l)\sqrt{1 - (kn + l)^2} + \sin^{-1}(kn + l)) + C \quad (16)$$

## 5.6 Decay Time: $w_i(n)$

The decay time for the percussion signal can be controlled through the amplitude envelopes  $w_i(n)$ . A natural sounding way to set these envelopes is to model them as exponentially decreasing envelopes over time:  $w_i(n) = A_0 e^{-n/\tau}$ , with different initial amplitude values  $A_0$ , as shown in Figure 10, and/or different decay rates  $\tau$ .

## 5.7 Commuted Synthesis Parameters

### 5.7.1 Attack Sharpness: Raised Cosine Envelopes

With raised cosines envelopes, small values of  $L$  create sharper sounding attacks, while longer values of  $L$  increase the presence of low frequencies in the output and result in bass-heavy sounds. Intuitively,  $L$  is proportional to the mass of a hammer or mallet used to excite a drum head: a longer  $L$  means a hammer/mallet with greater mass.

### 5.7.2 Attack Noisiness: Filtered Noise Bursts

For filtered noise burst excitations, a longer noise burst  $t_d$  and higher bandpass frequency cutoff  $f_{\text{high}}$  will create a noisier attack.  $f_{\text{low}}$  and  $f_{\text{high}}$  should be tuned to filter out undesired frequencies. For example, for a high pitched percussion sound, the lower frequencies could be filtered out from the noise burst by setting  $f_{\text{low}}$  to a higher frequency.

### 5.7.3 Timbre: Acoustic Resonator Impulse Response $r(n)$

The acoustic resonator filters the synthesis output, so the timbre can be further shaped by the frequencies present in  $r(n)$ . For an expansive and large sound, a room impulse response with a long T60 may work well while for a shorter, tuned sound, the impulse response of a small, acoustic tube model could be used.

## 6. SYNTHESIS EXAMPLES

While the loopback FM percussion synthesis method is capable of creating a variety of percussive sounds, this section covers three sound synthesis examples that use modal frequencies from [17]: the marimba, tom tom, and circular plate. Although these modal frequencies are associated with real, physical instruments, the aim of this synthesis is not to recreate the naturally occurring sounds. Rather, we seek to synthesize many different types of sounds with nonlinearities similar to those that occur in percussion instruments. For these examples, differences between percussion synthesis using traditional and loopback FM MS are compared for the same modal frequencies, decaying amplitude envelopes, and commuted synthesis parameters. Sound examples can be found at

<http://musicweb.ucsd.edu/~trsmyth/other/percussionSynthesisLoopbackFM.html>.

### 6.1 Marimba

Figure 9 compares the spectrograms of a marimba modeled as a bar with two free ends using traditional and loopback FM MS. This example sets  $\omega_{0,i}$  to seven modal frequencies



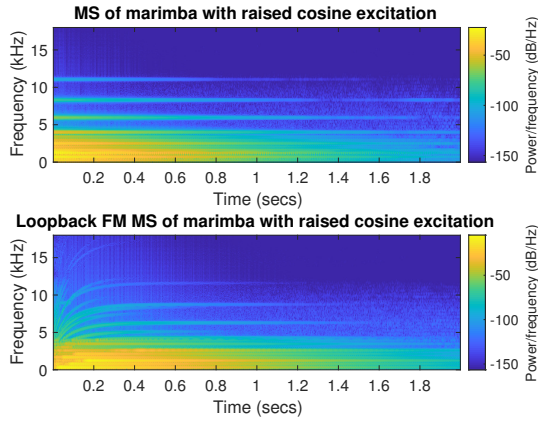


Figure 9. Traditional (top) vs. Loopback FM MS (bottom) using the modal frequencies of an ideal bar with two open ends. The excitation is a raised cosine and the acoustic resonator is an ideal tube synthesized using traditional MS.

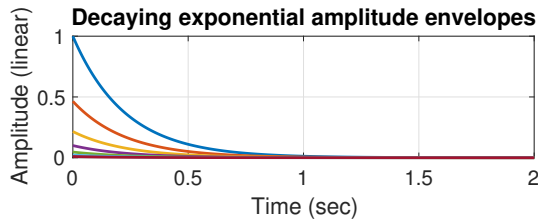


Figure 10. The marimba synthesis amplitude envelopes are decaying exponentials. The initial amplitude of the envelopes is inversely proportional to the modal frequency of the oscillator that is paired with the envelope.

calculated as

$$f_i = \begin{cases} 440, & \text{for } i = 0 \\ 440 \frac{(2i+3)^2}{3.011^2}, & \text{otherwise} \end{cases} \quad (17)$$

The amplitude envelopes are decaying exponentials where initial amplitudes decrease exponentially from 1 for the first (lowest) modal frequency to 0.01 for the seventh (highest) modal frequency. Figure 10 is a plot of the amplitude envelopes used for this marimba example. This example is created using  $z_{c,i}(n)$  oscillators with an 8-sample length raised cosine excitation and a pitch glide created by setting  $B(n) = 0.9999^n$ . The acoustic resonator is an ideal, open-closed tube synthesized using traditional MS. Compared to the signal generated using traditional MS, the signal created using loopback FM MS has more frequency components and a clearly increasing pitch glide.

## 6.2 Tom Tom

The spectrogram of a tom tom synthesized using traditional vs. loopback FM MS is shown in Figure 11. The modal frequencies used to synthesize the tom tom are

$$f_i = 142 \cdot [1, 2.15, 3.17, 3.42, 4.09, 4.80, 4.94] \quad (18)$$

The amplitude envelopes are the same as those used for the marimba as shown in Figure 10. The synthesis uses

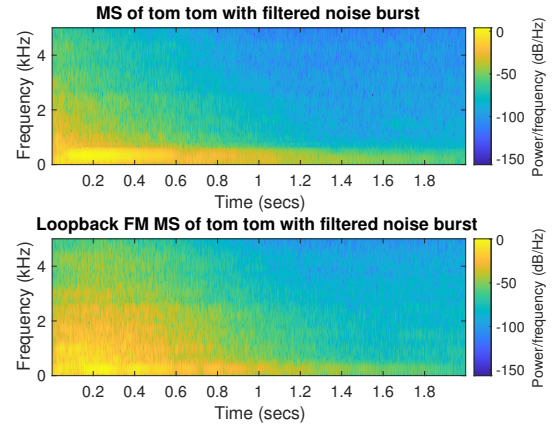


Figure 11. Traditional (top) vs. Loopback FM MS (bottom) using the modal frequencies of a tom tom. The excitation is a filtered noise burst and the acoustic resonator is a taiko drum recording.

$z_{0,i}(n)$  oscillators,  $b_0 = -0.9$ , and the pitch glide is created linearly increasing  $B$  from 0.55 to 0.91. The excitation is a 0.05 second long noise burst filtered with a 2<sup>nd</sup>-order Butterworth bandpass filter with frequency cutoffs at 120Hz and 4000Hz. The acoustic resonator is a recording of a taiko drum retrieved from [freesound.org](https://freesound.org). In Figure 11, there is more high frequency energy for the loopback FM MS than for the traditional MS, especially in the beginning of the signal.

## 6.3 Circular Plate

In Figure 12, loopback FM MS of a simply-supported circular plate is compared to traditional MS of the same circular plate. The modal frequencies used are

$$f_i = f_0 \cdot [1, 2.80, 5.15, 5.98, 9.75, 14.09, 14.91, 20.66, 26.99] \quad (19)$$

where  $f_0 = 0.2287c_L(h/a^2)$  for plate thickness  $h = 0.005\text{m}$ , plate radius  $a = 0.09\text{m}$ , and longitudinal wave speed  $c_L = \sqrt{E/\rho(1-\nu^2)}$  with Young's modulus  $E = 2 \cdot 10^{11}\text{N/m}^2$ , plate density  $\rho = 7860\text{kg/m}^3$ , and Poisson ratio  $\nu = 0.3$ . The amplitude envelopes are decaying exponentials over time. The initial amplitude of these envelopes decreases exponentially as frequency increases from 1 for the lowest modal frequency to 0.5 for the highest modal frequency. Using  $z_{c,i}(n)$  oscillators, a slight upwards pitch glide is created by linearly changing  $B(n)$  from 0.91 to 0.90 over the course of the signal. The excitation is an 8-sample long raised cosine envelope and the acoustic resonator is a room impulse response retrieved from [echothief.com](https://echothief.com). In this example, the drastic aliasing effects in loopback FM MS are used to create an extremely “noisy” signal. Perceptually, the traditional MS output sounds like a clean bell sound, while the loopback FM MS sounds more like a noisy, struck cymbal.

## 7. CONCLUSIONS

This work has presented a real-time method to synthesize novel, abstract percussion sounds using MS with loopback

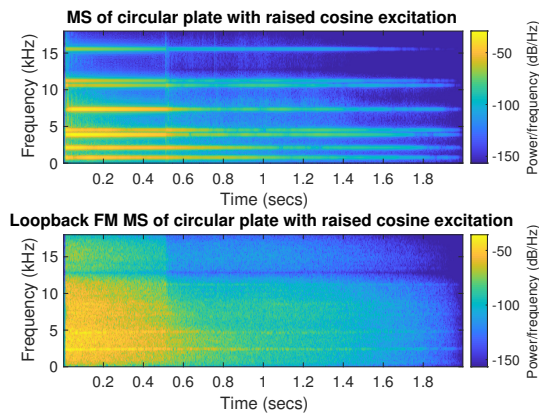


Figure 12. Traditional (top) vs. Loopback FM MS (bottom) using the modal frequencies of a simply-supported circular plate. The excitation is a raised cosine, and the resonator is a room impulse response.

FM oscillators. Loopback FM creates complex spectra and pitch glides similar to the nonlinear effects observed in existing percussion instruments. The synthesis technique allows for parametric control of musical dimensions including sounding frequency, decay time, timbre, and pitch glide. Synthesis examples using the modal frequencies of a marimba, tom tom, and circular plate are examined.

A future research direction involves investigating the aliasing that occurs with large carrier frequencies for both loopback FM and its closed form expression. Another research interest is to explore other methods of creating nonlinearities in oscillators and using these methods with loopback FM to create an even wider range of percussion sounds.

### Acknowledgments

This research was inspired by a conversation with Professor Miller Puckette when he shared with us his implementation of Loopback FM in Pure Data (Pd). We are immensely grateful to Miller for the time and insight that he offered to make this work possible.

## 8. REFERENCES

- [1] M. Ducceschi and C. Touzé, “Modal approach for nonlinear vibrations of damped impacted plates: Application to sound synthesis of gongs and cymbals,” *Journal of Sound and Vibration*, vol. 344, pp. 313–331, Jan. 2015.
- [2] F. Avanzini and R. Marogna, “A modular physically based approach to the sound synthesis of membrane percussion instruments,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 891–902, May 2010.
- [3] S. Bilbao, “A family of conservative finite difference schemes for the dynamical von Karman plate equations,” *Numerical Methods for Partial Differential Equations*, vol. 24, no. 1, pp. 193–216, Jan. 2008.
- [4] S. A. V. Duyne and J. O. Smith, “Physical modeling with the 2-d digital waveguide mesh,” in *Proceedings of the International Computer Music Conference*, Tokyo, Japan, 1993, pp. 40–47.
- [5] N. H. Fletcher, “The nonlinear physics of musical instruments,” *Reports on Progress in Physics*, vol. 62, pp. 723–764, 1999.
- [6] S. Petrausch and R. Rabenstein, “Tension modulated nonlinear 2d models for digital sound synthesis with the functional transformation method,” in *Proceedings of EUSIPCO-05, Thirteenth European Signal Processing Conference*, Antalya, Turkey, Sept. 2005.
- [7] S. Bilbao, “Sound synthesis for nonlinear plates,” in *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx’05)*, Madrid, Spain, Sept. 2005.
- [8] R. Marogna, F. Avanzini, and B. Bank, “Energy based synthesis of tension modulation in membranes,” in *Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*, Graz, Austria, Sept. 2010.
- [9] M. Jossic, D. Roze, and T. Hélie, “Energy shaping of a softening duffing oscillator using the formalism of port-hamiltonian systems,” in *Proceedings of the 20th International Conference on Digital Audio Effects (DAFx’17)*, Edinburgh, UK, Sept. 2017.
- [10] T. Smyth and J. Hsu, “On phase and pitch in loopback frequency modulation with a time-varying feedback coefficient,” in *26th International Congress on Sound and Vibration*, Montreal, Canada, July 2019.
- [11] —, “Representations of self-coupled oscillators with time-varying frequency,” in *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.
- [12] J. M. Adrien, “The missing link: Modal synthesis,” in *Representations of Musical Signals*. The MIT Press, 1991, pp. 269–295.
- [13] N. Tomisawa, “Tone production method for an electronic musical instrument,” U.S. Patent 4 249 447A, Feb 1998.
- [14] V. Välimäki, J. S. Abel, and J. O. Smith, “Spectral delay filters,” *Journal of the Audio Engineering Society*, vol. 57, no. 7/8, pp. 521–531, July/Aug. 2009.
- [15] J. O. Smith, “Efficient synthesis of stringed musical instruments,” in *Proceedings of the International Computer Music Conference*, Tokyo, Japan, Sept. 1993.
- [16] W. G. Gardner, “Efficient convolution without input-output delay,” *Journal of the Audio Engineering Society*, vol. 43, no. 3, pp. 127–136, March 1995.
- [17] T. Rossing, *Science of Percussion Instruments*, ser. Series in Popular Science. Singapore: World Scientific, 2000, vol. 3.

# Deep Linear Autoregressive Model for Interpretable Prediction of Expressive Tempo

Akira Maezawa

Yamaha Corporation

akira.maezawa@music.yamaha.com

## ABSTRACT

Anticipating a human musician's tempo for a given piece of music using a predictable model is important for interactive music applications, but existing studies base such an anticipation based on hand-crafted features. Based on recent trends in using deep learning for music performance rendering, we present an online method for multi-step prediction of the tempo curve, given the past history of tempo curves and the music score that the user is playing. We present a linear autoregressive model whose parameters are determined by a deep convolutional neural network whose input is the music score and the history of tempo curve; such an architecture allows the machine to acquire a music performance idioms based on musical contexts, while being able to predict the timing based on the user's playing. Evaluations show that our model is capable of improving the tempo estimate over a commonly-used baseline for tempo prediction by 18%.

## 1. INTRODUCTION

When multiple musicians play in a music ensemble for the first time, each player responds to one another by listening to each other and anticipating each others' timing. Realizing this kind of on-the-fly online timing prediction for machines is important for interactive computer systems such as automatic accompaniment systems, since the system needs to respond in real-time in spite of the delays in computer systems and/or mechanical actuators, which can be on the order of hundreds of milliseconds [1].

In this kind of problem setting, key requirements are (1) awareness to the common musical idioms associated with a particular music score, (2) awareness to how the human performer has executed the playing, and (3) interpretability of the system behavior. Awareness to the music score is important because the music score and expressive parameters are highly correlated [2]. For example, musicians often slow down before the end of the song. Awareness to the actual performance by the human musician is also important because, as much as the music score provides strong cues on musical idioms, it is the performer who ultimately chooses to abide by or defy it. Interpretability is important

because it allows musicians to anticipate how the system will respond to their playing.

Recently, it was shown in the analysis of duet interaction [3] that using hand-crafted features from the music score and the performance helps in timing prediction, as opposed to using the performance features alone. However, it uses simple hand-crafted features from the music score, potentially limiting the kind of information captured from the music score. To bypass design of handcrafted features, in a different problem of music performance rendering, deep learning has shown promise for acquiring features that are relevant for the prediction of note strengths [4].

Inspired by these works, this paper presents a tempo prediction method that takes into account both the music performance context and the surrounding music score context, and learns the feature representation in a data-driven manner<sup>1</sup>. This is achieved by training a linear autoregressive (AR) model of the tempo, whose coefficients are generated from a deep neural network (DNN) that takes both the performance history and musical context as the inputs.

Our contributions are as follows:

1. We propose deep linear AR model, a linear AR model whose coefficients are modeled with a DNN.
2. We apply the deep linear AR model for human tempo prediction, allowing online tempo prediction that is both music performance-aware and music context-aware.
3. We evaluate our model, comparing it with other commonly-used baseline methods for human tempo prediction in interactive music systems and show that DNN-based feature extraction surpasses hand-crafted features. Furthermore, through application of performance rendering, we shed light on the kind of musical context the system learns to acquire through the DNN.

Audio examples of the inferences made by our method is available at <https://sites.google.com/view/deep-linear-ar-for-tempo/>.

## 2. RELATED WORK

### 2.1 Automatic accompaniment

Predicting the human player's tempo is a critical component in automatic accompaniment systems [5–8]. To tune

<sup>1</sup> In this paper, we use the word “tempo” interchangeably with the beat duration.

the prediction to a particular performer that plays a particular piece, such a system often learns a model of tempo curve from multiple rehearsals. Timing prediction through rehearsals, however, is agnostic to musical contexts, so it is not possible to predict the expressive timing on a piece that has never been rehearsed before. While it is possible to use tempo markings written in the music score [9], but it is often cumbersome to prepare such an annotation. This paper is concerned with enabling the machine to anticipate expressive timing on a piece that has not been played before, or to respond to spontaneous musical ideas for pieces that have been rehearsed.

## 2.2 Music performance rendering

Music performance rendering method generates a human-like tempo curve, given a previously unseen music score [10, 11]. It is critical in this task to extract features from the music score that are relevant to music performance, a reign in which deep learning has shown promise, particularly for predicting note strengths [4] and timings [12]. Unfortunately, predicting and responding to live human performance is outside the scope of the problem definition. This paper is concerned with using an external tempo curve played by a human musician to predict the tempo curve, using a model that is amenable to online inference.

## 2.3 Duet interaction

Duet interaction [3], the task of predicting the machine response given a human playing in a human-machine ensemble, exploits the music score to improve the quality of timing and dynamics prediction with a few number of rehearsals. A limitation is that the method requires hand-crafted features from the music score. This paper is concerned with using the idea of duet interaction for human timing prediction.

## 2.4 Deep non-linear AR models

Recently, deep neural networks have been applied to sequence prediction tasks, where non-linear AR model [13] has shown success. However, its behavior is often difficult to predict ahead of time. In real-time systems like automatic accompaniment, it is desirable for the system to exhibit a known dynamics ahead of time, using models like linear autoregressive models [5] for which stability and sensitivity is easy to analyze. This paper is concerned with generating a mathematical model based on linear autoregressive process, so that the behavior of the system for a given piece of music can be anticipated beforehand, while enjoying the high-level feature design that deep learning offers.

## 3. OUR METHOD

The goal of our method is to predict the tempo curve of a musician who plays a new piece of music score, as if a group of musicians are anticipating each other's timing for a piece that they play for the first time. We require *multi-step* predictions: at a given time instance when the user

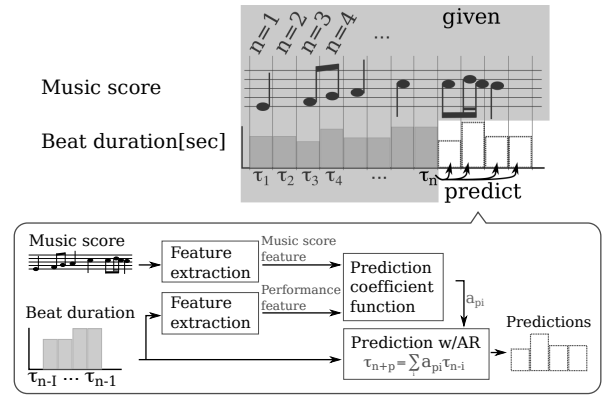


Figure 1: Overview of our method. Our method, given the music score and the history of beat duration, predicts the successive durations. It uses a DNN that generates AR model coefficients to predict the beat duration from past beat durations.

is playing some position  $n$  in the music score, we predict the tempo at  $n$  plus  $p$  eighth notes, ranging from  $p = 1$  to  $p = P$ . This way, it is possible to predict the future playing position for various interactive systems with different latencies.

As shown in Figure 1, the music score is assumed to be segmented at an eighth-note level, where the  $n$ th segment is associated with a segment duration  $\tau_n$ . When the player has just finished playing the  $n$ th segment, our goal is to predict the future segment durations,  $\tau_{n+1}$  to  $\tau_{n+P}$ , given the music score and the segment durations played by the player up to now,  $\{\tau_{n'}\}_{n' \leq n}$ .

Our method predicts the timing with an AR model of order  $I$ . The AR coefficients are determined by two inputs. First, since it is autoregressive, it uses the segment duration history of the current performance,  $\{\tau_{n'}\}_{n' \leq n}$ . Second, since the music score and the tempo are highly correlated, it uses the music score information around the current segment  $n$ , which we denote by  $S_n$ . It contains (1) the notes written in the score, *i.e.*, the pitches, the start times and the durations, and (2) metric information, *i.e.*, the meter and the relative position inside the measure.

### 3.1 Deep linear AR model for timing prediction

We formulate the timing prediction as a multi-step prediction problem. Suppose that the performer has played just up to segment  $n$ . We assume that the expected segment duration  $\tau_{n+p}$  ( $p > 0$ ) depends only on the performance history  $\tau_{n' \leq n}$  and the music score  $S_n$ . Furthermore, we assume that the residual follows a zero-mean Laplacian noise with scale  $\lambda$ . We assume Laplacian noise because it is tolerant to outliers of the IOI. Then, based on the assumptions described later in Section 3.1.1, we can formulate timing prediction as a maximum likelihood estimation of the following probabilistic model:

$$\tau_{n+p} | \Theta, S_n, \tau_n \sim \mathcal{L} \left( \sum_{i=0}^{I-1} a_{p,i} (S_n, \tau_n; \Theta) \tau_{n-i}, \lambda \right), \quad (1)$$



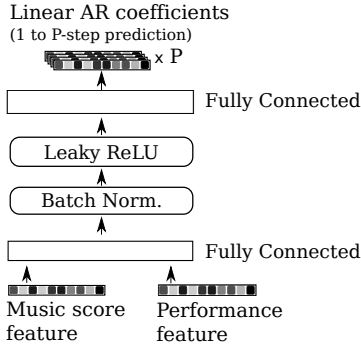


Figure 2: The prediction coefficient function. It is a NN that, given the music score feature and the performance feature, generates the AR model parameters of order  $I$ , for up to  $P$ -step prediction.

where  $\mathcal{L}(\mu, \lambda)$  is a Laplace distribution with the position parameter  $\mu$  and the scale parameter  $\lambda$ ,  $\Theta$  denotes a set of arbitrary model parameters, and  $\tau_n = [\tau_{n-I+1} \cdots \tau_n]$ . Our goal is to design the non-linear function  $a_{p,i}(S_n, \tau_n; \Theta)$ , which we call the **prediction coefficient function**, and to learn its model parameters  $\Theta$ .

We represent the prediction coefficient function  $a_{p,i}$  as a neural network composed of two fully-connected layers as shown in Figure 2. The number of neurons for the hidden layers is 300 and the number of output neurons is  $P \times I$ . It uses leaky rectified linear units (ReLU) for the activation function and each layer is batch-normalized [14].

The inputs to the network are low-dimensional feature representations of the music score  $S_n$  and the performance  $\tau_n$ . We denote these features respectively by  $u_n$  and  $v_n$  and call them the **music score feature** and **performance feature** respectively.

### 3.1.1 Derivation of the deep linear AR model

We assume that at segment  $n$ , only the previous  $I$  coefficients contribute to the estimate of  $\tau_{n'+n}$ . Then, we model  $\tau_{n+p}$  as the following non-linear AR process:

$$\tau_{n+p} = f_p(S_n, \{\tau_{n'}\}_{n' \leq n}; \Theta) + \epsilon_{n,p}; \epsilon_{n,p} \sim \mathcal{L}(0, \lambda). \quad (2)$$

To make the model's behavior more predictable, which is beneficial for real-time interactive music applications,  $f_p$  is approximated by a first-order Taylor expansion with respect to  $\tau$  to yield the following:

$$\tau_{n+p} \approx \tau_n^T (\nabla_{\tau} f_p(S_n, \tau; \Theta))|_{\tau=0} + H(S_n, \tau_n; \Theta) + \epsilon_{n,p}, \quad (3)$$

where  $H$  is the higher-order term left-divided by  $\tau_n^T$ , and we assumed that the constant term is zero<sup>2</sup>. Thus, we arrive at Equation 1, where  $a_{p,i} = (\nabla_{\tau} f_p(S_n, \tau; \Theta))|_{\tau=0} + H(S_n, \tau_n; \Theta)_i$ .

<sup>2</sup> Incorporating the constant term yielded in poor results in preliminary experiments.

### 3.1.2 Relationship with linear AR and deep non-linear AR models

Our model is a compromise between a linear AR model used in automatic accompaniment systems [5] and a deep non-linear AR model used in areas like speech generation [13]. It is a linear AR process, whose model parameters are governed by a non-linear function  $a(\cdot)$ .

Our modeling approximation is inspired by the success of shortcut connections in deep learning [15, 16]: our model can be thought of as having a multiplicative shortcut connection from the input  $\tau_n$  to the output, so that the output gradient is able to fully exploit the input.

### 3.2 DNN for feature extraction

For computing the music score feature  $u_n$ , we extract the following attributes from the music score  $S_n$ :

1.  $\phi_n^{(1)} \in \{0, 1\}^{12}$ : Denotes the downbeat phase; it is a one-hot vector that indicates, at segment  $n$ , the number of segments that have elapsed since the last downbeat.
2.  $\phi_n^{(2)} \in \{0, 1\}^{12}$ : Denotes the meter; it is a one-hot representation of the meter at the current measure, expressed as the number of segments inside a measure, with the longest meter of 12/8.
3.  $\phi_n^{(3)} \in \{0, 1\}^{128 \times 20}$ : Denotes the notated notes; it is a binary piano-roll representation of the music score between segment  $n-2$  and  $n+2$ ; the piano-roll is quantized at 32nd-note level, and the pitch is represented as MIDI note number between 0 and 127.

Given these data, we extract the music score feature using a DNN shown in Figure 3. Namely,  $\phi^{(1)}$  and  $\phi^{(2)}$  are concatenated and passed through a fully-connected layer to obtain an intermediate feature  $\phi^{(m)}$ .  $\phi^{(3)}$  is passed through three convolutional layers followed by a fully-connected layer to obtain another intermediate feature  $\phi^{(p)}$ . We use leaky ReLU for activation, followed by batch-normalization and max-pooling. The convolutional layers and max-pooling layers are designed so that the network becomes (1) sensitive to particular harmonic progressions or note patterns, (2) sensitive to position in the score, and (3) relatively invariant to transposition. Specifically, for the first layer, we attempt to capture interval relationship by using kernel size of twelve semitones by two 32nd notes. Furthermore, to achieve invariance on transposition while remaining sensitive to the temporal positions, max-pooling is done only on the pitch axis, spanning four semitones. To obtain the music score feature  $u_n$ , we concatenate these intermediate features from the current measure and  $W$  neighboring segments,  $\{\phi_{n'}^{(m)}, \phi_{n'}^{(p)}\}_{n'=n-W}^{n+W}$ . By evaluating from  $n-W$  up to  $n+W$ , we incorporate both prior and upcoming contexts, both of which are relevant for musical expression [17].

For the performance feature  $v_n$ , we use  $\tau_n$ , normalizing it to have zero mean and unit variance. Thus, at the expense of ignoring the dependency of average tempo on tempo expression [18], it expresses the local trend of the

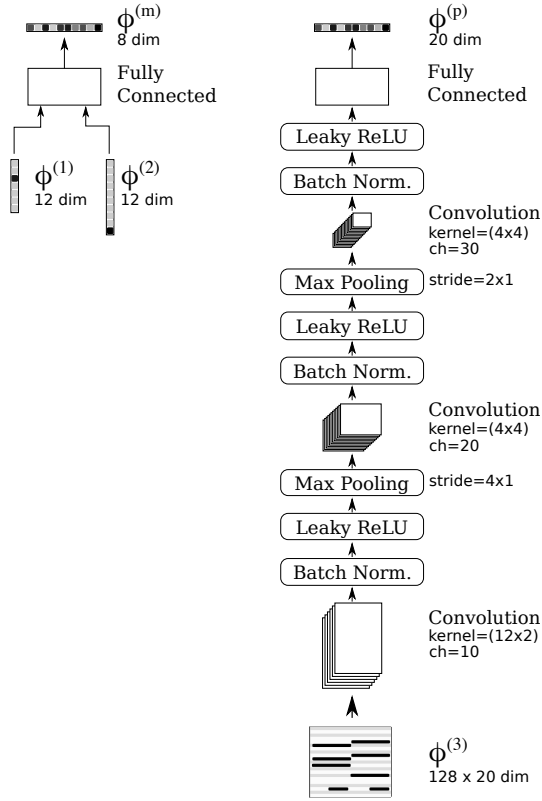


Figure 3: DNN for music score feature extraction.

tempo change, while being invariant under the change of the average tempo at segment  $n$ .

### 3.3 Training

In our method, we train the parameters related to DNN for music score feature extraction, and the parameters related to the prediction coefficient function  $a_{p,i}(u_n, v_n; \Theta)$ . We train the parameters as to maximize the likelihood of the ground-truth segment duration, which amounts to minimizing the  $l_1$  loss with respect all the parameters, accumulated over all the songs in the training data. Specifically, for each song in the training data, the loss is given as follows, where  $N$  is the number of segments in the song,  $\hat{\tau}_n$  is the ground-truth segment duration, and  $\hat{S}$  is the music score:

$$\sum_{n=I}^N \sum_{p=1}^P |\hat{\tau}_{n+p} - \sum_{i=0}^{I-1} a_{p,i}(u_n(\hat{S}), v_n(\hat{\tau}_n)) \hat{\tau}_{n-i}|. \quad (4)$$

#### 3.3.1 Data augmentation strategies

Since the segment duration is expected to be invariant under transposition, we augment the data by randomly shifting the piano-roll  $\phi^{(3)}$  by -5 to +5 semitones. Furthermore, we simulate the motor noise of a human musician, inspired by models of sensorimotor synchronization [19]; we add to  $\tau_n$  a correlated Gaussian noise  $e_n$ , given as  $e_n = \epsilon_n - \epsilon_{n-1}$  where  $\epsilon_n$  is a white noise with a standard deviation of 10 ms. Time-stretching, a common data-

augmentation strategy for audio [20], was not used, because our model is invariant under change of the average tempo in a piece.

## 4. EVALUATION

To evaluate our method, we conducted three experiments. First, we evaluated the effectiveness of incorporating the music score and the performance history for tempo prediction. Second, we conducted an ablation study for assessing the effect of using a deep convolutional neural network for music score feature extraction. Third, we qualitatively analyzed the typical predictions made by our model, by applying our model for tempo curve generation.

In the subsequence experiments, we let  $P = 8$ ,  $I = 24$ , and  $W = 24$ . To train the model, we used ADAM [21] for seven epochs with a batch-size of 128, with the same hyper-parameters used in [21]. We directly minimized the loss function, with no pre-training.

### 4.1 Dataset

We evaluated our method on 52 virtuoso solo pieces played by different people, mostly pieces from the Romantic era such as Chopin, Liszt, Schubert, and virtuosic Beethoven piano sonatas<sup>3</sup>. The pieces were chosen because they often contain extreme tempo fluctuations, owing to the high freedom allowed in playing.

First, for each piece, a digital music score was prepared as a standard MIDI file. Second, performance data for each piece was obtained from Yamaha e-Piano competition, which contains performances by different performers on a Yamaha Disklavier player piano to record the MIDI performance data (up to sixteen interpretations per piece). We obtained a total of 250 MIDI performance data. Finally, for each MIDI performance data, the ground-truth segment durations and the music score were obtained by aligning it to the corresponding music score MIDI data. The alignment was obtained by using a symbolic alignment method, followed by a manual inspection by a trained musician.

Of the 52 pieces, we used 47 pieces for training and 5 for testing, using ten-fold cross validation (about 712,000 training samples).

### 4.2 Evaluation of the prediction method

In this experiment, we evaluated the effectiveness of using the performance feature and the music score feature. To this end, we have evaluated the prediction error of multi-step prediction. Specifically, for each prediction step  $p$ , we computed the mean absolute error of the predicted duration between the current segment  $n$  and segment  $n + p$ , and the actual duration. We have evaluated the prediction errors using the following methods:

1. Condition **MA**:  $\tau_{n+p}$  is predicted as a moving average of  $\tau_{n-I}$  to  $\tau_{n-1}$ , similar to [22].

<sup>3</sup> A full list of the repertoire is available at the web page mentioned in the introduction.

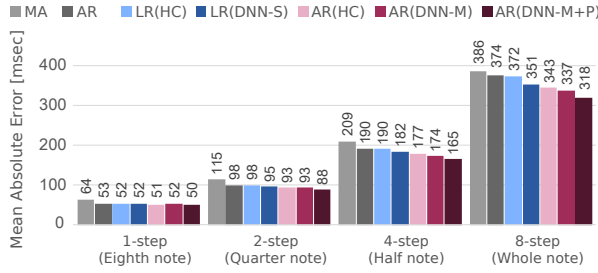


Figure 4: Mean absolute error of multi-step prediction with different baselines. Differences are statistically significant for all pairs of the baseline and the proposed method (Kruskal-Wallis Test applied pairwise,  $p < 0.01$ ).

2. Condition **AR**:  $\tau_{n+p}$  is predicted with a single  $AR(I)$  model, similar to [5].
3. Condition **LR(HC)** (HC=Hand-Crafted):  $\tau_{n+p}$  is predicted with linear regression, using as the input variables the timing history and hand-crafted features used in duet interaction. Specifically, the hand-crafted features consists of the segment durations, the beat phase and the coefficients to the quadratic regression of the highest and lowest notes. This condition amounts to a simplified model of duet interaction [3], where the player's performance is used to predict his own timing. To make a fair comparison on music score feature extraction, we omit features that are not obtainable from the music score and the beat duration history, such as the note strengths.
4. Condition **LR(DNN-M)**: Same as **LR(HC)**, except instead of hand-crafted features, the DNN-based music score feature is used.
5. Condition **AR(HC)**: Same as **LR(HC)**, except instead of using linear regression on hand-crafted features, autoregressive coefficients are directly estimated from the hand-crafted features using the DNN described in Figure 2.
6. Condition **AR(DNN-M)**: The proposed method without the performance feature. It uses a DNN-based score feature extraction, and a DNN-based autoregressive coefficient extraction.
7. Condition **AR(DNN-M+P)**: The proposed method. It uses both performance and a DNN-based score features, and a DNN-based autoregressive coefficient extraction.

#### 4.2.1 Results and discussion

The results are shown in Figure 4. First, the proposed method consistently outperforms the baseline methods, by up to 18% when compared with MA for 8-step prediction.

Second, prediction with an AR model outperforms a LR model, for both hand-crafted and linear features. That is, AR(HC) outperforms LR(HC) and AR(DNN-M) outperforms LR(DNN-M). This is surprising because linear regression is, in our context, auto-regression with an additional bias term explained by the features. This shows that

auto-regressive models without a bias term is beneficial for expressive tempo prediction.

Third, feature extraction with DNN surpasses hand-crafted features, for both linear regression and auto-regressive models. That is, LR(DNN-M) outperforms LR(HC) and AR(DNN-M) outperforms AR(HC). This shows that directly training feature extraction from a symbolic music information is beneficial for expressive tempo prediction.

Finally, the incorporation of the performance history  $\tau$  improves the prediction. That is, AR(DNN-M+P) outperforms AR(DNN-M). The effect is more prominent when making a prediction with a long forecast like a half note ahead (4-step) or a whole note ahead (8-step).

#### 4.2.2 Distributions of the prediction errors

The distribution of 8-step prediction error is shown in Figure 5. For sake of clarity, we only show distributions that highlight properties of different features or prediction models.

It first shows that moving average (MA), the simplest method of all, is unbiased but suffers from the worst outlier. This is reasonable because it is good at tracking steady tempo, but has no capability to anticipate the next tempo during tempo changes.

Second, AR model tends to make negative errors, *i.e.*, it tends to anticipate that the next note will slow down. Such a tendency arises because musicians tend to slow down more often in a given song than they speed up. This kind of asymmetric tempo change encourages the AR model to anticipate every note to be slowing down when using squared error for training.

Finally, the proposed method enjoys increased robustness against outliers. Therefore the primary benefit of incorporating performance feature is the capability to prevent a large mistake.

### 4.3 Evaluation of music score feature extraction

We have conducted an ablation study to assess which components are effective for computing the music score feature. To this end, we have compared the multi-step prediction errors when using different kinds of music score feature extractor as follows:

1. Condition **FC**: The music score feature is obtained using one fully-connected layer applied to the piano-roll. Combined with the two fully-connected layers in the prediction coefficient function, this model is a perceptron with three hidden layers, making it similar in essence to the architecture used for generation of expressive dynamics from the music score [4].
2. Condition **Conv1**: The music score feature is obtained by one convolutional layer followed by max-pooling and fully-connected layer.
3. Condition **Conv2**: Same as Conv1, except we use two convolutional layers.

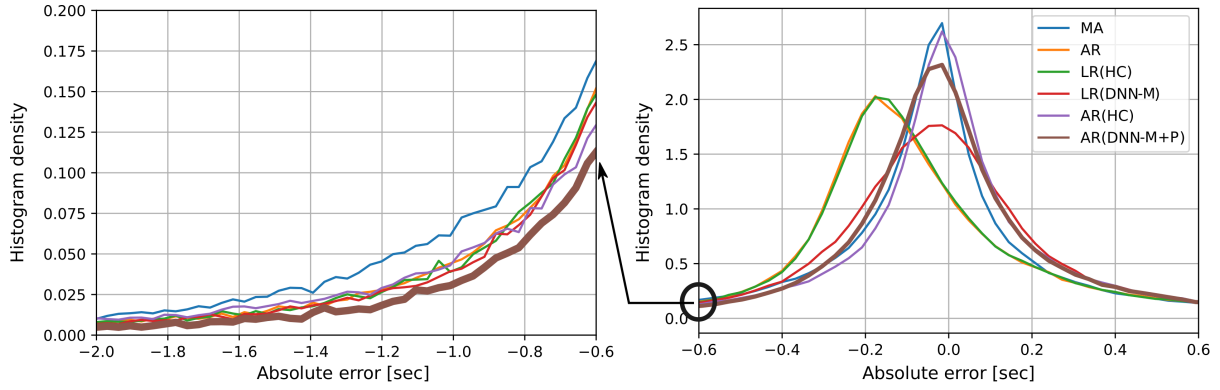


Figure 5: Histograms of the prediction errors, centered about the origin (right), and zoomed in for tails (left).

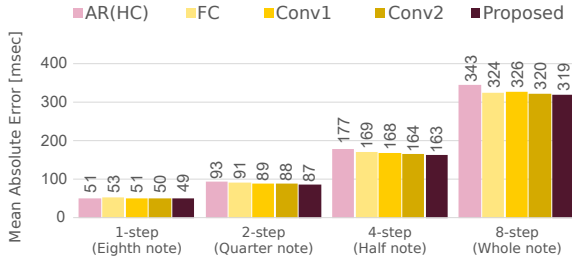


Figure 6: Multi-step prediction error with different ways of computing the music score feature. Differences of absolute errors are significant, except for condition FC and Conv1 of 2-step prediction and FC and Conv2 of 8-step prediction (Kruskal-Wallis Test,  $p < 0.01$ ).

#### 4.3.1 Results and discussion

The results are shown in Figure 6. It can be seen that even with simple network like FC, it helps to automate feature extraction, as can be seen by comparing with AR(HC). Furthermore, addition of more convolutional layers improves the accuracy.

### 4.4 Analysis through performance rendering

To qualitatively analyze the kind of prediction made by the model, we analyzed the tempo curve generated by the model when it predicts the tempo based on its own predictions. That is, instead of predicting the tempo curve based on a human performance, we drove the AR model with its 1-step prediction, and apply the following low-pass filter to let the prediction stay about some average  $m$ :

$$\tau_n = (1 - \alpha)m + \alpha \sum_{i=0}^{I-1} a_{1,i}(S_n, \tau_n; \Theta) \tau_{n-i}. \quad (5)$$

Here,  $\alpha \in [0, 1]$  is a parameter that controls how much  $\tau_n$  reverts to  $m$ .  $m$  was set to the mean tempo for each piece, and  $\alpha$  was set to 0.5.

#### 4.4.1 Results and discussion

In Figure 7, we present two examples from songs not contained in the training dataset, one demonstrating a performance idiom pertaining to harmony and another specific to

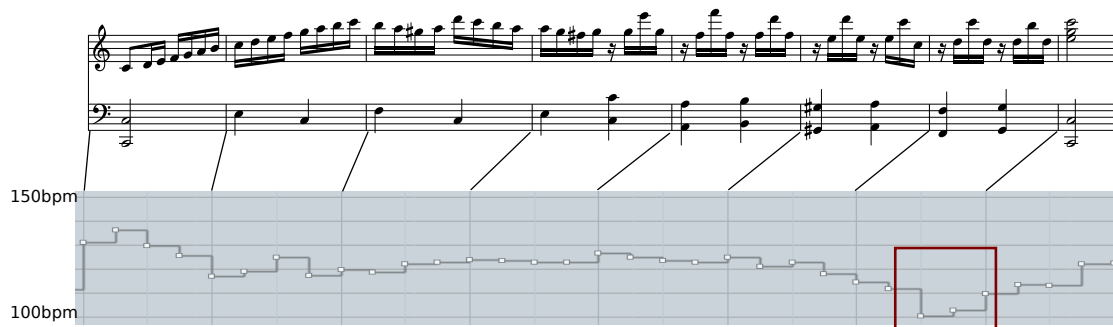
piano playing. We invite the readers to listen to the examples at the web page mentioned in the introduction.

First, the method seems to capture performance idioms related to cadences. To demonstrate, Figure 7a shows an excerpt from a simple piece, Mozart’s Variations on Twinkle Twinkle Little Star. The generated tempo slows down in the bounded rectangle, which is a perfect cadence. It shows that the method is capable of capturing a common idiom of slowing down before a cadence. This behavior is quite consistent and also seen in other variations as well.

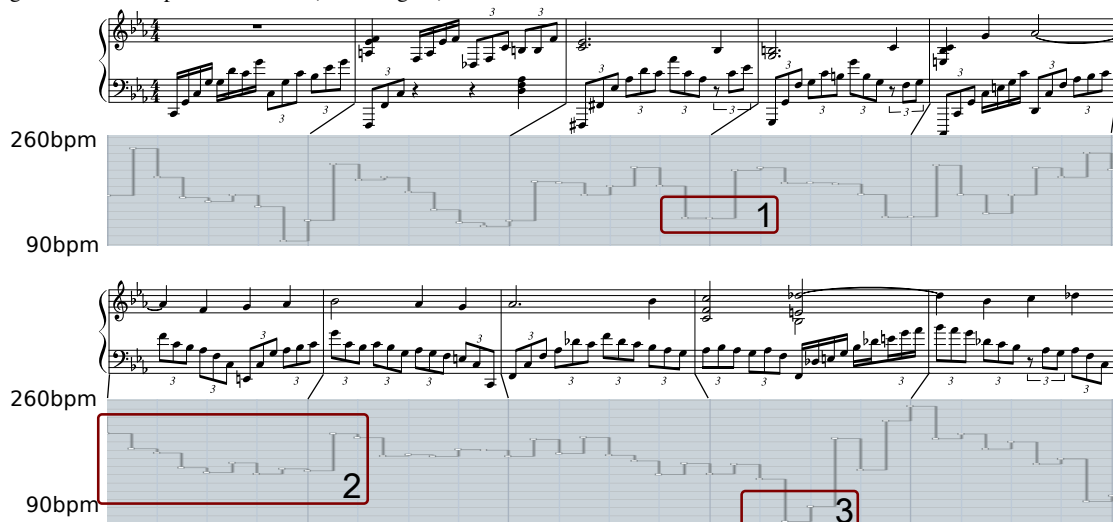
Second, the method seems to also capture a typical idiom pertaining to the left-hand technique. Figure 7b shows an excerpt from a technically and harmonically more complex piece, Rachmaninoff’s Piano Concerto No. 2. We observe, qualitatively, a few idioms related to piano playing. First, the beginning of the bar tends to start slowly, forming an arc-like tempo curve (label “1” in Figure 7b). This kind of playing is consistent with a typical piano playing [2,23]. Second, this kind of behavior is not hard-coded, but rather depends on the surrounding musical context; for example, the tempo does not slow down in a non-cadence progression (label “2,” a progression from D dim7 to C7/E, which will resolve to Fm). Furthermore, the most prominent drop in the tempo occurs at a climactic segment inside the phrase (label “3”). These behavior seem concordant with how this particular piece is played.

The generated tempo curve depends on harmonic changes or accompaniment patterns, suggesting that the music score feature extraction was able to learn relevant relationship between music score and tempo changes. We believe that the kind of predictions made by our method captures the essence of music context and performance for making sensible timing predictions in interactive music systems. These results show the expressiveness of our model, despite the fact it uses far fewer information from the score than those typically used in music performance rendering [11], but they also qualitatively address some limitations of our method. First, it does not take into account the genre. The generated tempo curves are mostly in the style of late Romantic pieces which tend to exaggerate the tempo, but sometimes such exaggerated tempo curves are stylistically inappropriate for earlier music like the Mozart example. Second, the system is agnostic to the larger structural con-





(a) Example for Mozart's Variations on Twinkle Twinkle Little Star, K. 265, variation 7. The model seems to acquire the idiomatic slowing-down before a perfect cadence (boxed region).



(b) Example for Rachmaninoff's Piano Concerto No. 2, Mvt. 1, measure 52-61. The model seems to acquire the idiomatic arc-like tempo curve (denoted 1), but the behavior is dependent on surrounding context (denoted 2). The most dramatic drop of tempo occurs at  $Fm \rightarrow E \dim 7/F$  progression (denoted 3).

Figure 7: Examples of the generated tempo curves.

text. For example, the Mozart example shows the  $A_1$  section to a variation whose structure is ternary, *i.e.*,  $A_1BA_2$ . The system consistently slows down the last cadence, but it is generally appropriate to only slow down the  $A_2$  section. Third, the method is agnostic to (1) additional cues in the music score, such as the phrase, the expression and the tempo markings, and (2) performance cues like the articulation and the dynamics.

#### 4.4.2 On the ease of stability analysis

It is easy to analyze and modify the behavior of our model since we model the prediction as a linear AR model, whose properties are well-understood. We believe that such a capability to analyze and correct the system's behavior is beneficial for real-time interactive music applications such as automatic music accompaniment, since it provides an interpretable form of performance guarantee for human musicians.

To demonstrate, we have trained our model and estimated the AR coefficients at one point for a given music score  $S_n$  and past performance history  $\tau_{n-i}$ , for 1-step tempo prediction. Figure 8 shows the poles and the frequency response of the inferred AR process. It can be seen, for

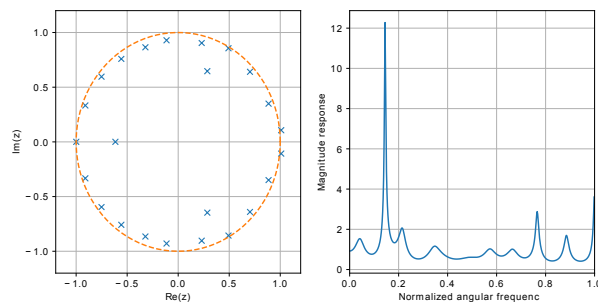


Figure 8: Pole-zero diagram (left) and the frequency response (right) of the autoregressive model inferred using our method.

example, that the system is unstable because the poles are outside the unit circle, resonating to oscillations at a normalized frequency of 0.15, or about a dotted eighth note. If a stable behavior is desired, then it is possible to correct the AR coefficients such that the maximum magnitude response is bounded.

## 5. CONCLUSION

This paper proposed an online method for predicting a human performer's expressive timing, based on the music score and the performance history. The method is both music score-aware and performance-aware, and is capable of extracting useful features from the score that are relevant to timing prediction. We have shown on a difficult dataset of expressive virtuoso piano playing that (1) incorporating both contextual information from the performance and the music score contributes to accurate timing prediction, (2) a deep architecture, especially convolutional architecture, is useful for extracting relevant features from the music score, and (3) the model seems to acquire common idioms in piano playing, according to the generated tempo curves.

Future work includes (1) integrating the model with interactive music systems, (2) predicting more aspects of human music performance like the dynamics, (3) incorporating of more elements of the music score like the dynamics, phrasing and expressive marking, and (4) incorporation of additional performance cues such as the dynamics and body gestures for prediction.

## 6. REFERENCES

- [1] A. Maezawa and K. Yamamoto, "MuEns: A multi-modal human-machine music ensemble for live concert performance," in *Proc. CHI*, 2017, pp. 4290–4301.
- [2] C. Palmer, "Music performance," *Annual review of psychology*, vol. 48, no. 1, pp. 115–138, 1997.
- [3] G. Xia, Y. Wang, R. B. Dannenberg, and G. Gordon, "Spectral learning for expressive interactive ensemble music performance," in *Proc. ISMIR*, 2015, pp. 816–822.
- [4] S. Van Herwaarden, M. Grachten, and W. B. De Haas, "Predicting expressive dynamics in piano performances using neural networks," in *Proc. ISMIR*, 2014, pp. 45–52.
- [5] S. Wada, Y. Horiuchi, and S. Kuroiwa, "Temo prediction model for accompaniment system," in *Proc. ICMC*, 2014, pp. 1298–1303.
- [6] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proc. ICMC*, 1984, pp. 193–198.
- [7] A. Cont, "Antescofo: Anticipatory synchronization and control of interactive parameters in computer music," in *Proc. ICMC*, 2008, pp. 33–40.
- [8] C. Raphael, "A Bayesian network for real-time musical accompaniment," in *Proc. NIPS*, 2001, pp. 1433–1439.
- [9] F. Krebs and M. Grachten, "Combining score and filter based models to predict tempo fluctuations in expressive music performances," in *Proc. SMC*, Copenhagen, Denmark, 2012.
- [10] K. Okumura, S. Sako, and T. Kitamura, "Laminae: A stochastic modeling-based autonomous performance rendering system that elucidates performer characteristics," in *Proc. ICMC*, 2014.
- [11] G. Widmer, S. Flossmann, and M. Grachten, "YQX plays Chopin," *AI Magazine*, vol. 30, no. 3, p. 35, 2009. [Online]. Available: <https://www.aaai.org/ojs/index.php/aimagazine/article/view/2249>
- [12] M. Grachten and C. Cancino Chacon, *Temporal Dependencies in the Expressive Timing of Classical Piano Performances*, 04 2017, pp. 360–369.
- [13] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," in *Arxiv*, 2016. [Online]. Available: <https://arxiv.org/abs/1609.03499>
- [14] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456. [Online]. Available: <http://dblp.uni-trier.de/db/conf/icml/icml2015.html#IoffeS15>
- [15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.
- [16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: <http://dx.doi.org/10.1162/neco.1997.9.8.1735>
- [17] E. Istok, A. Friberg, M. Huotilainen, and M. Tervanieniemi, "Expressive timing facilitates the neural processing of phrase boundaries in music: Evidence from event-related potentials," *PLOS ONE*, vol. 8, no. 1, pp. 1–11, 01 2013. [Online]. Available: <https://doi.org/10.1371/journal.pone.0055150>
- [18] H. Honing, "Is expressive timing relational invariant under tempo transformation?" *Psychology of Music*, vol. 35, no. 2, pp. 276–285, 2007. [Online]. Available: <https://doi.org/10.1177/0305735607070380>
- [19] B. H. Repp, "Sensorimotor synchronization: a review of the tapping literature," *Psychonomic bulletin & review*, vol. 12, no. 6, pp. 969–992, 2005.
- [20] B. McFee, E. J. Humphrey, and J. P. Bello, "A software framework for musical data augmentation," in *Proc. ISMIR*, 2015.
- [21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, 2014. [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [22] R. Yamamoto, S. Sako, and T. Kitamura, "Robust online algorithm for real-time audio-to-score alignment based on a delayed decision and anticipation framework," in *Proc. ICASSP*, May 2013, pp. 191–195.
- [23] D. Stowell and E. Chew, "Bayesian MAP estimation of piecewise arcs in tempo time-series," in *Proc. CMMR*, 2012.

# METRICS FOR THE AUTOMATIC ASSESSMENT OF MUSIC HARMONY AWARENESS IN CHILDREN

**Federico Avanzini, Adriano Baratè,  
Luca A. Ludovico**

LIM – Laboratorio di Informatica Musicale  
Department of Computer Science  
University of Milan  
name.surname@unimi.it

**Marcella Mandanici**

Department of Music Education  
Music Conservatory “L. Marenzio”  
Brescia (Italy)  
mmandanici@gmail.com

## ABSTRACT

In the context of a general research question about the effectiveness of computer-based technologies applied to early music-harmony learning, this paper proposes a web-based tool to foster and quantitatively measure harmonic awareness in children. To this end, we have developed a web interface where young learners can listen to the leading voice of well-known music pieces and associate chords to it. During the activity, their actions can be monitored, recorded, and analyzed. An early experimentation involved 45 school teachers, whose performances have been measured in order to get user-acceptance opinions from domain experts and to determine the most suitable metrics to conduct automated performance analysis. This paper focuses on the latter aspect and proposes a set of candidate metrics to be used for future experimentation with children.

## 1. INTRODUCTION

Tonal harmony can be defined as an idiom, or system of rules, which “... governs how melodies and chords are organized throughout the duration of a tonal musical composition” [1, p.194]. Systematically defined by Rameau [2], tonal harmony has been employed in various musical styles, spanning from the Baroque period to contemporary popular songs [1,3]. It has been demonstrated that both children and adults who are not musicians have a strong feeling for harmony and are able to recognize the tonic chord [4], implicit harmonies [5], and chord progressions [6]. In this sense, the development of a harmony awareness extends beyond the boundaries of formal music education.

Many treatises and handbooks [7–11] describe the tonal system in terms of keys, chords and scales. Even though these concepts represent useful tools for analysis and theoretical pedagogy, they fail to explain the perceptual qualities of chord relationships that are peculiar to the tonal system [1]. Such a formal pedagogical approach has been severely criticized by some scholars [12], and it has been

held responsible for failure and disaffection towards harmony studies. Eberlein, in particular, complains about the use of rules that are in contrast with practice and perception and about the absence of explicit stylistic references in tonal harmony treatises [13]. Therefore, it is not surprising that pedagogical approaches to tonal harmony have developed little outside professional music curricula, and that little attention has been paid to harmony education programs for children, high school students, and amateurs.

The work presented in this paper is part of an ongoing research on computer-based technologies applied to music-harmony learning. We have recently proposed a web-based tool that implements a set of experiences focused on harmonic skills and awareness form primary and middle school students [14]. Particularly for primary school children the use of the web interface should be complemented with a series of perceptual and physical activities - i.e. musical games - which focus on some fundamental concepts related to tonal harmony. The games, thoroughly described in [15] constitute the basis for understanding the tasks required in the various groups of experiences presented in the web interface. However in this context the current goal is to devise suitable metrics for objectively assessing children performance in the experiences, based on the recordings and analysis of their actions.

Objectively assessing musical abilities is a much studied – and controversial – problem. Musical aptitude batteries proposed in the second half of 20th century are now considered obsolete in several respects [16]. The concept of musical ability is multifaceted and includes various types of musical capacity (e.g., tempo, pitch, rhythm, timbre, melody perception) that are not easily separated. Building on previous research such as [17] and [18], more recent music games for education [19] have been developed. Commercial systems such as Smart Music<sup>1</sup> and Yousician<sup>2</sup> offer gamified approaches to music instrument learning, and academic research focuses on the development of objective descriptors for assessing music performances [20]. Although these works are mainly concerned with musical instrument performance rather than theoretical music abilities, they share some common traits with the experiences proposed in our web tool, namely a performative dimension and a gamified approach. As far as con-

Copyright: © 2019 Federico Avanzini, Adriano Baratè, et al.  
This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> <https://www.smartmusic.com/>

<sup>2</sup> <https://www.yousician.com/>

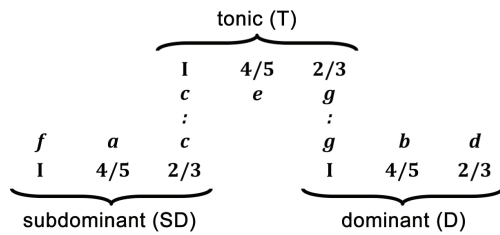


Figure 1. Chart of the relationships of the three tonal functions, also known as primary chords; picture adapted from [25, p. 7].

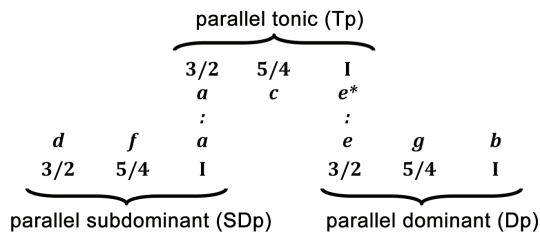


Figure 2. Chart of the relationships of the three parallel chords; picture adapted from [25, p. 7].

cerns the study of tonal harmony many computer interfaces and systems have been designed to help the understanding of musical chords and harmonic progressions [21–24], included *Mapping Tonal Harmony*, an innovative tool for visualizing the various shifts through harmonic regions in real time.<sup>3</sup> However all these systems are rather complex to use and are not finalized for use in primary or middle schools.

## 2. THEORETICAL BACKGROUND

Following Riemann's theory of tonal harmony [25], we divide the tonal space into primary and parallel chords. The primary chords are tonic (**T**), dominant (**D**) and subdominant (**SD**); the parallel chords are parallel tonic (**Tp**), parallel dominant (**Dp**) and parallel subdominant (**SDp**). Figure 1 depicts the three tonal functions **T**, **D**, and **SD** (primary chords, namely I, V, and IV degree). The Pythagorean relationships reported below the pitches show the origin of the major harmonic functions and chords starting from *c* (tonic).

Riemann considered the minor chords as the product of the inversion of the harmonic series. Thus, he derived the remaining chords (II, III, and VI degree) from the reversed harmonic series starting from *e* (marked with an asterisk in Figure 2). This is an abstract scheme from which much more complex harmonies can be derived. However, it can fit a number of popular songs as well as classic music harmonization patterns which can be a good starting point for understanding harmonic functions.

Building on this theoretical background, we designed a simple activity to be conducted with primary school children in order to assess and possibly improve their aware-

ness towards tonal harmony. In general terms, we ask students to complete a number of tasks consisting in associating a single chord to each music tune. Chords are selected from the Riemann's scheme. Tunes are chosen (and sometimes modified) to best fit a single chord, typically the tonic one; this implies that all the notes of the leading voice occurring on beats belong to the pitches forming that chord.

But is there one right choice and five wrong options? Basically, the answer is: no. There is a more plausible chord in terms of tonal harmony, since the proposed music tunes are built on the notes forming the tonic chord. Besides, the themes selected for the experience are well known to listeners who are used to link them to a given harmonic accompaniment. Nevertheless, other options are possible. First, some of the proposed chords share one or more notes with the expected one (e.g., the minor triad on the VI grade has two notes in common with the major triad on the I grade), so a music tune insisting on the I and III grade of a major scale could be harmonized by a VI-grade triad as well, with no conflicting notes among the leading voice and the accompaniment. Moreover, chords forming seventh, ninth and even more complex intervals are common in many musical genres, so accompanying a tonal music tune through unconventional chords would be perfectly acceptable, and arguably also more interesting to some listeners.

To ease the navigation of the harmonic space and to enhance the perceptual differences between the primary chord zone (all major chords) and the parallel chord zone (all minor chords), we propose to place the chords along a circle with the tonic, dominant and subdominant in the lower part of the circle and the parallels in front of their relatives, as depicted in Figure 3 (white arrows).

## 3. THE WEB TOOL

*Harmonic Touch* is a Web platform for the study and practice of tonal harmony.<sup>4</sup> This application is conceived as a step-by-step wizard that leads users through three experiences towards the discovery of important features of tonal harmony by leveraging on chord perception, gestural interaction and gamification techniques.

<sup>4</sup> <http://didacta18.lim.di.unimi.it/eng/>

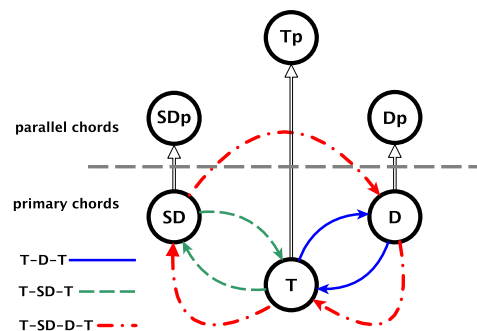


Figure 3. The spatial arrangement of primary and parallel chords with the route of three common chord progressions

<sup>3</sup> <https://mdecks.com/mapharmony.phtml>



*Harmonic Touch* implements a step-by-step process towards harmony awareness based on the administration of three groups of experiences, focusing on: 1) the recognition of the implicit harmony, 2) the timed recognition of harmonic changes, and 3) melody harmonization. These three types of experiences have been discussed in detail in [14]. In this context, we will focus on the first group of experiences. To this end, users are asked to match a short music tune with a single chord that, in their opinion, best fits the whole melody. The chord must be selected from the six primary and parallel chords discussed above.

This type of experience can be proposed to learners in a physical space, namely moving across different spots on a floor, as well as through a web-based interface that simulates the mentioned setting. The former approach implements a real embodiment of a harmonic path, whereas the latter presents advantages in terms of ease of use and computer-based performance, for both the leading voice and underlying chords. A mixed approach is also possible, thus applying to music education the principles of *algorithmicity* learning methodology defined in [26].

In the Web interface, available chords are represented as shown in Figure 4, with positions randomly rotated and no explicit indication of tonal functions. These are made explicit in the visual representation of the harmonic space of the third group of experiences, where the user has to explore the harmonic space to find the chords for melody harmonization (see Figure 5). Understanding the relative layout of chords is left to the user, who can explore them freely during an initial training phase. This spatial arrangement carries some important peculiarities:

- The user can get acquainted with the sound of the various chords by simply clicking or touching a set of buttons that follow this arrangement: an important facility for people who cannot play a polyphonic instrument or for children [27];
- The user can intuitively couple the chord qualities with their relative location, which can help the memorization of the sound of the various chords as well as the routes of the most important harmonic progressions, as shown by the colored arrows in Figure 3.

#### 4. RESEARCH QUESTIONS

There is a clear distinction between the research questions that brought us to develop *Harmonic Touch* and to test it in different contexts on one side, and those investigated in this paper on the other.

Concerning the former questions, the general goal of the project is to engage learners through playful physical and computer-supported activities in a topic often considered too abstract and difficult for young students or amateurs. As mentioned before, harmony awareness is trained firstly by making the user recognize the implicit harmony in a music tune, then focusing on the timed detection of harmonic changes in a theme, and finally inviting the learner to perform melody harmonization (i.e., selecting a sequence

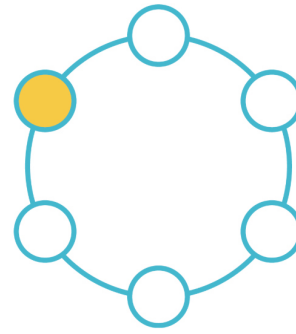


Figure 4. The interface for the recognition of the implicit harmony of a music tune. No indication is provided to the user about harmonic functions and their spatial disposition, that can be reconstructed only through exploration.

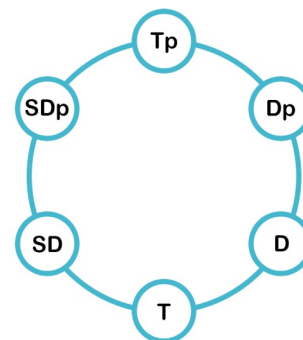


Figure 5. The interface for melody harmonization with explicit indications of the harmonic functions employed in the third group of experiences.

of chords and playing them at the right time while listening to a tune). The research questions addressed by the *Harmonic Touch* project deal with the efficacy of an embodied approach to harmony learning and with the educational support offered by technologically enhanced tools.

In this paper, conversely, we focus on a specific aspect: how can we analyze the large amount of data recorded during the experimentation phase, with respect to the first group of experiences only?

From a terminological point of view, in the following we will call *test* each experience completed by a user, *session* the group of experiences performed by a single user, and *activity* the collection of sessions completed by all users. Consequently, the assignment for each test is: “Associate one chord out of the six available to this music tune”; the assignment for a user session is: “Complete 4 tests”. Music tunes have been proposed in the same order to all participants.

During each test, users could:

- play the leading voice with chord accompaniment (action *P*);
- rewind the music tune and play it back as many times as they wanted (action *R*);
- make subsequent choices selecting the current chord



Figure 6. Users experiencing *Harmonic Touch* at Didacta 2018.

from the mentioned circle, and keep it playing under the leading voice (actions  $C_1 \dots C_n$ );

- confirm the final choice and exit (action  $X$ ).

In order to measure the abilities in the recognition of implicit tonal harmony, actions  $P$ ,  $R$ ,  $C_n$ , and  $X$  have been recorded user by user for each test, thus generating a considerable set of information to be investigated along different axes.

## 5. GATHERING THE DATA SET

*Harmonic Touch* was presented to primary and middle school teachers in occasion of the 2<sup>nd</sup> edition of Didacta Italy, Florence, October 18-20, 2018.<sup>5</sup> This initiative is the most important Italian fair focusing on education, vocational training and relation among school and work.

Specifically, *Harmonic Touch* was presented during a workshop on music education and digital languages, with the aim to involve music teachers in the use of the interface (see Figure 6). But, as a side effect, we were interested also in analyzing teachers' performances, consequently we tracked their results on a set of increasingly difficult harmony experiences and profiled them through anonymous questionnaires administered before and after the experimentation in order to obtain more accurate indications from tests. Finally, we also gathered user observations and suggestions about content and technology-related issues, through a set of questions about the whole experience.

The workshop was attended by an audience of 45 educators, mainly teaching in middle (57%) and in primary school (25%). Sixteen percent of the audience were males; mean age = 49.8 (median = 51); mean working age = 22.7 years (median = 20 years). The employed melodies are all well-known tunes, as confirmed by 44 participants out of 45. The complete data set of answers to pre-activity and post-activity surveys is publicly available.<sup>6</sup>

Concerning user performances for the first activity, i.e. the battery of 4 tests for the recognition of implicit har-

mony, results are publicly available, too.<sup>7</sup> For each test, 2 diagrams are available (see Figure 7):

1. The path that was followed in the chord circle. In this diagram, positions are reported in the same form (e.g.,  $T$  is always the bottom chord) in order to ease the comparison of paths, but remember that users experienced random rotations for each test;
2. The time of chord changes, including clicks on the same chord. Time scale is the same for all users, with the exception of User 25 who was discarded from time scale calculation due to the very long time employed in the answer. This diagram carries also information about actions  $R$  and  $X$ . The former action was often invoked by users since, at the end of the piece, music stopped with no looping. This diagram presents a dark background when music is playing; chord changes occurring on a white background are chords played after the end of the melody.

Blank spaces indicate tests with no saved results, as the complete session of User 6 or single tests of Users 8, 16, 31, etc.

## 6. ANALYZING THE DATA SET

The experimentation at Didacta 2018 was not conducted on the intended final users for such an experience, who should be primary school students. In this sense, gathered data cannot be analyzed to assess the pedagogical effectiveness of the proposed approach. Rather, the goal of the analysis is to discover and fine-tune the most suitable metrics to extract significant information about user awareness of tonal harmony.

### 6.1 Analysis Dimensions

Going back to the definitions of *test*, *session* and *activity* provided in Section 4, we can identify the following dimensions to analyze:

- Horizontal axis – The goal is to track, user by user, the behavior and improvements from one test to another. This effort can bring to the identification of well-defined user profiles, e.g. the “frantic explorer” or the “self-confident listener” (see below);
- Vertical axis – The purpose is to recognize similar behaviors across users when approaching the same test. An aspect evaluated vertically is the average time to complete the  $n$ -th test, or the distribution of answers about the final chord for a given test;
- Global dimension – The data collected during the whole activity, namely in each test of each session, are evaluated globally in order to assess general aspects. Prototypical behaviors may emerge from a joint analysis of the horizontal and the vertical axes.

<sup>5</sup> <http://fieradidacta.indire.it/>

<sup>6</sup> [http://www.lim.di.unimi.it/data/didacta\\_survey/](http://www.lim.di.unimi.it/data/didacta_survey/)

<sup>7</sup> [http://didacta18.lim.di.unimi.it/results/index\\_results.html](http://didacta18.lim.di.unimi.it/results/index_results.html)

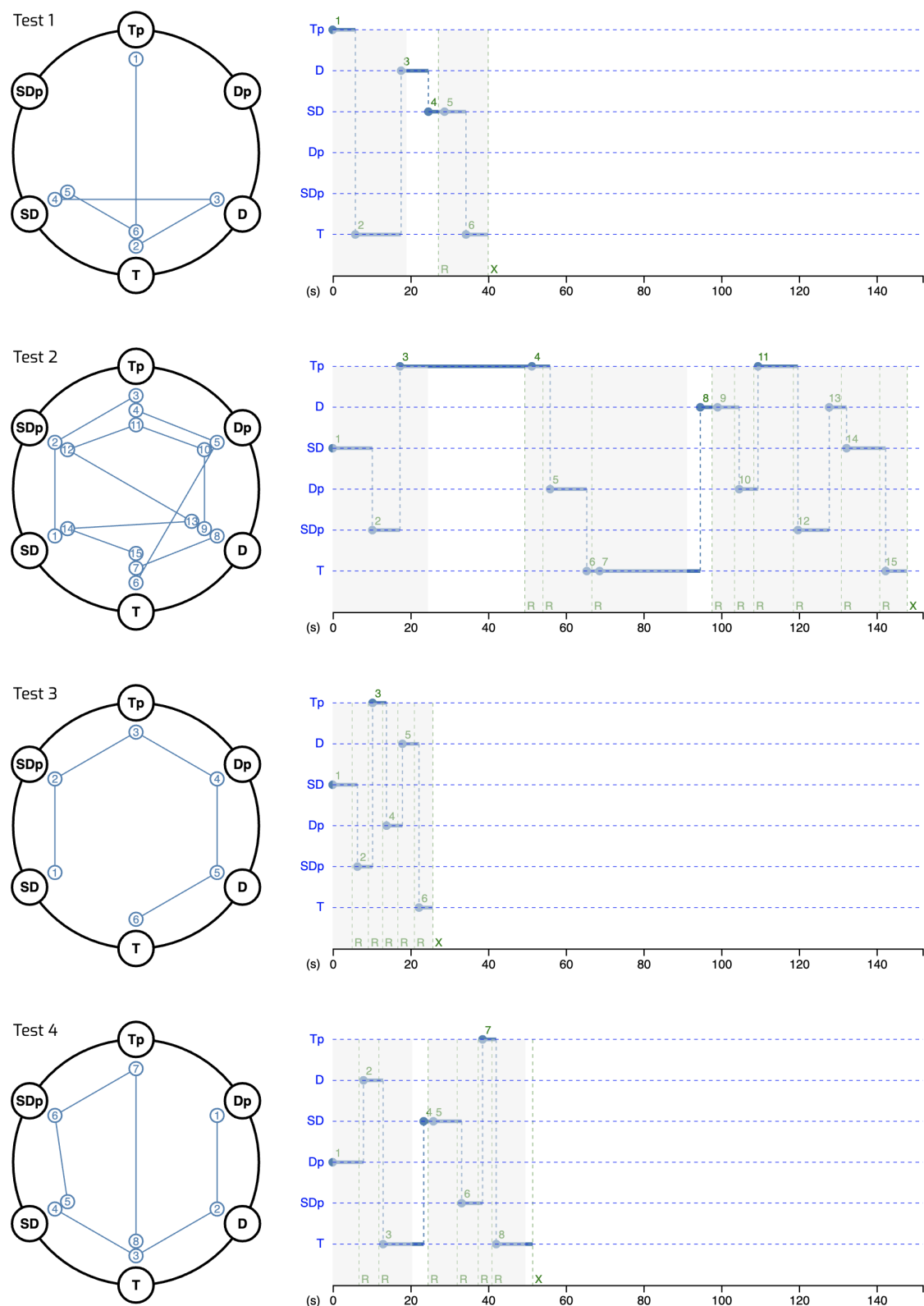


Figure 7. Diagrams tracking the performances of a randomly selected user (35). The left part of the image represents the chord circle with the path followed by the user. In the right part information about the times of chord changes is provided, including clicks on the same chord

Concerning the horizontal axis, first it is possible to compare the duration of each test in a session. The expected effect is a decreasing amount of time to determine the final chord. In the analysis of the total time spent on each test, it is important to underline that music themes have different lengths (19, 25, 17, and 9 seconds respectively), and present increasingly difficult harmonic situations. In any case, results do not confirm the initial prediction, and the time spent on each test does not follow a common trend. For the horizontal axis, one of the most interesting aspects to highlight is the evolution of the path followed by single users from test to test. Since the relative chord positions were kept unaltered across the activity, we could expect a learning effect. The first choice of each test can be considered random, since circles are rotated; but, after a training phase, previous experience should bring users to quickly jump to the desired chord. Conversely, from test to test we often noticed the replication of an exploratory behavior. The learning effect, if any, pushed some users to stop at the first occurrence of the tonic chord, or to minimize the number of attempts after that, but not to jump towards the final destination with confidence. In this sense, one of the rare counterexamples is the performance of User 1 in Test 4, which was the last one of the session: after randomly performing action  $C_1$  and selecting **SD**, in about 5 seconds he/she moved to **T** through action  $C_2$  and stopped.

The vertical axis investigates the common characteristics of each test. For example, it is possible to compute: the mean and the variance of the total amount of time spent by users on a given test; the distribution of the chords selected as the final choice; the number of chords listened before fixing the last one; and so on. Considering common behaviors of users test by test, the first experience, presented as a trial to get acquainted with the interface, has been considered exploratory by most of them. In this case, a number of wrong behaviors emerged, for example the choice of new chords with no music playing; conversely, later tests showed a general improvement in performances. Another behavior common to many users when passing from early tests to the last ones is the tendency to restart the piece more. In this way, users can compare the fitness of each chord against the first notes of the leading voice, limiting the experience to the initial part of the tune, whereas, at the beginning, most of them played the entire piece and changed chords while music was advancing. Since these actions can be found in many users, such a consideration brings us to the global dimension.

Concerning the whole activity, we evaluated the average time spent on each chord selection, test by test and user by user. The expected learning effect should bring to a gradual decrease of listening times, and, actually, this is noticed in many user sessions. Moreover, the analysis makes another aspect emerge: users tended to rest on the chord they considered correct, even in the middle of the experience, namely before confirming it as the final choice.

The analysis of dark-colored areas in time diagrams often justifies the repeated selection of the same chord: this happens when music stops and clicking on a new chord does not restart the performance, thus users manually invoke ac-

tion  $R$  and immediately re-select the chord. For instance, this effect occurs in Test 1 (actions  $C_4, C_5$ ), Test 2 (actions  $C_3, C_4$ ; actions  $C_8, C_9$ ), and Test 4 (actions  $C_4, C_5$ ) in Figure 7.

Finally, a comprehensive analysis of the whole activity lets us determine mainly two well-defined user profiles:

1. the “frantic explorer”, who wanders around the circle, usually following a clockwise or counterclockwise path and going on until the whole circle has been covered, even multiple times. Although this behavior could be associated to a casual way to explore, actually we did not retrieve many examples of random patterns;
2. the “self-confident user”, who stops almost immediately after choosing the expected chord.

In this specific case, it would be interesting to study the correlation between self-declared music skills and all the aspects we have mentioned before, e.g.: the final choice, the total time spent on each test, the number of trials before selecting the final chord, the average listening time for each chord, and so on. Nevertheless, please remember that such an experience has been conceived for young learners, who probably have no prior music education.

## 6.2 Metrics

Concerning the metrics to automatically assess the educational efficacy, a first point regards the learning curve. This aspect can be investigated through different numeric values: the amount of time and/or the number of trials to find the final chord after its first listening, the number of waypoints to reach the expected chord after listening to the first one. For each of the mentioned values, we can consider the minimum, maximum, mean and variance.

Another indicator regarding the learning curve is the evolution of aggregated times spent on each chord, both the selected one and those considered wrong by the user. The distribution of final choices could also redefine the concept of right vs. wrong chords, even if in a tonal context it is possible to rank chords on the base of the pitches in common with the leading voice.

For each of the mentioned aspects, we can investigate the horizontal as well as the vertical axis, thus focusing on user-specific or test-specific analysis respectively. Global considerations emerge from a comprehensive insight involving both dimensions. Figure 8 shows the distribution of chords selected by users at the end of each test. The convergence of answers towards the tonic chord **T** seems to demonstrate the existence of a harmonic awareness, at least in the context of tonal harmony. The distributions of results in the first three tests (all pieces in major key and taken from music literature) are very similar, with **T** chosen by about 80% of the participants; the fourth test (in minor key) differs from the previous ones, with **T** chosen by only 57% of users.

Table 1 illustrates some additional metrics computed on the Didacta 2018 sample. Let  $c$  be the chord selected by the user at the end of the test, and  $t_0(c)$  the time when



Table 1. Minimum  $m$ , maximum  $M$ , mean  $\mu$ , and standard deviation  $\sigma$  of elapsed time and number of explored chords.

	Time				Chords			
	$m$	$M$	$\mu$	$\sigma$	$m$	$M$	$\mu$	$\sigma$
Test 1	0.55	211.65	65.01	49.38	0	22	5.78	5.49
Test 2	1.25	577.71	75.18	90.82	0	27	7.66	5.82
Test 3	1.03	185.91	38.43	38.32	0	18	4.73	4.53
Test 4	2.06	174.36	43.38	38.00	0	32	7.18	7.56

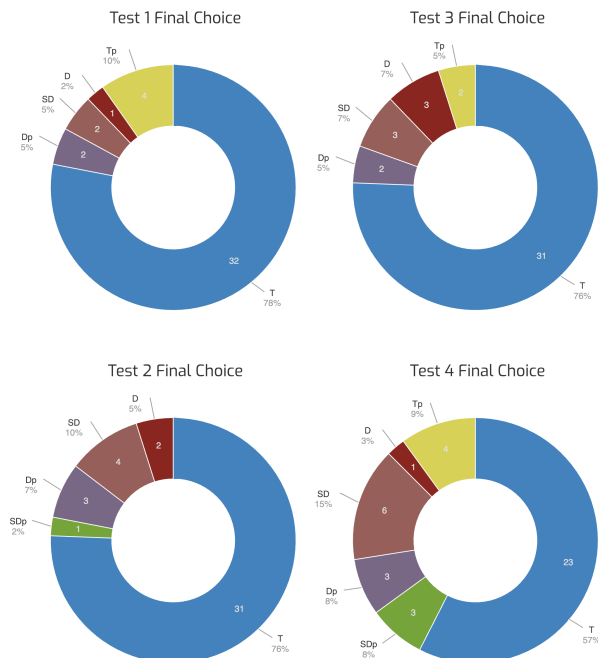


Figure 8. Distribution of the chords selected at the end of each test.

$c$  is first clicked,  $t_1(c)$  the time when the user confirms its final choice; the table shows the minimum, maximum, mean, and standard deviation of  $t = t_1(c) - t_0(c)$  and of the number of selections occurred between time  $t_0(c)$  and  $t_1(c)$ . An analysis of such values on one side demonstrates a tendency towards an exploratory behavior by users, who go on navigating through chords also after listening to the one they consider correct; on the other side, the evolution of times across tests shows a trend of decreasing, which could imply an improvement in self-confidence and harmonic awareness. We can expect that a more noticeable learning effect may emerge only after a number of administrations and may be observed across a long timespan.

A global indicator of the harmonic awareness is necessarily based on multiple metrics, sometimes coming from already available data, sometimes involving additional features that we will implement in the future. For example, in the current version of *Harmonic Touch* there is no way to assess consistency, namely to test coherence of user choices when a music tune is administered multiple times.

Even if the computation of values for the mentioned indicators could be easily extracted from the data reported above, it would make little sense to provide this kind of

information on the sample involved at Didacta 2018, since domain experts could behave very differently from young learners.

## 7. CONCLUSIONS

The main goal of this work was paving the way for the analysis of big amounts of data coming from an extensive experimentation of *Harmonic Touch*, to be conducted on primary school children instead of domain experts. In this sense, the Didacta 2018 experience was the occasion to present the educational approach and test the functionality of the software framework.

A critical analysis of gathered data, little significant with respect to the educational valence, allowed us to identify different dimensions to be investigated when the experimentation campaign will involve the expected final users, i.e. children. To each dimension we associated a number of indicators useful to assess the educational activity.

As a side effect of the experimentation occurred at Didacta 2018, some change requests emerged, both in the general pedagogic approach and in the implementation. Concerning the former, teachers asked the possibility to customize contents and to adapt them to their educational goals. In addition, the corpus of experiences should provide a more gradual evolution from very clear tonal situations to more complex pieces. The familiarity of children with music themes may also have an impact on user performances, so this aspect could be better assessed in the future.

As for implementation issues, some recurrent wrong behaviors will push us to improve the usability of the interface. An example is to restart music when a new chord is selected and the leading voice is no more playing, instead of explicitly request the user to push the Play button.

## 8. REFERENCES

- [1] D. Butler and H. Brown, "Describing the mental representation of tonality in music," in *Musical Perceptions*, R. Ajello and J. Sloboda, Eds. Oxford University Press, 1994.
- [2] J. P. Rameau, *Traité de l'harmonie reduite à ses principes naturels; divisé en quatre livres. Livre I. Du rapport des raisons & proportions harmoniques. Livre II. De la nature & de la propriété des accords et de tout ce qui peut servir à rendre une musique parfaite. Livre III. Principes de composition. Livre IV. Principes d'accompagnement.* Imprimerie de Jean-Baptiste-Christophe Ballard. A Paris, 1722.

- [3] S. G. Laitz, *The complete musician: An integrated approach to tonal theory, analysis, and listening*. Oxford University Press New York, 2012.
- [4] E. G. Schellenberg, E. Bigand, B. Poulin-Charronnat, C. Garnier, and C. Stevens, “Children’s implicit knowledge of harmony in western music,” *Developmental Science*, vol. 8, no. 6, pp. 551–566, 2005.
- [5] L. J. Trainor and S. E. Trehub, “Key membership and implied harmony in western tonal music: Developmental perspectives,” *Perception & Psychophysics*, vol. 56, no. 2, pp. 125–132, 1994.
- [6] K. A. Corrigan and L. J. Trainor, “Effects of musical training on key and harmony perception,” *Annals of the New York Academy of Sciences*, vol. 1169, no. 1, pp. 164–168, 2009.
- [7] N. Rimsky-Korsakov and J. Vītolis, *Practical manual of harmony*. C. Fischer, 1930.
- [8] C. Kœchlin, *Traité de l’harmonie: en 3 volumes*. Eschig, 1928.
- [9] A. Schoenberg, *Theory of harmony*. Univ. of California Press, 1983.
- [10] W. Piston, *Harmony*. WW Norton, 1948.
- [11] S. M. Kostka, D. Payne, and B. Almén, *Tonal harmony*. New York, 1984.
- [12] É. Jaques-Dalcroze, “Petite histoire de la rythmique,” *Le rythme*, vol. 39, pp. 3–18, 1935.
- [13] R. Eberlein, “A method of analysing harmony, based on interval patterns or “gestalten”,” in *Music, gestalt, and computing*. Springer, 1997, pp. 225–236.
- [14] M. Mandanici, F. Avanzini, A. Baratè, and L. A. Ludovico, “A computer-based approach to teach tonal harmony to young students,” in *Proc. 11th Int. Conf. Computer Supported Education (CSEDU 2019)*, 2019.
- [15] M. Mandanici, L. A. Ludovico, F. Avanzini, and A. Baratè, “Learning tonal harmony through bodily interactions and gamification,” in *Proc. 9th Conf. European Network of Music Educators and Researchers of Young Children (MERYC2019)*, 2019.
- [16] L. N. Law and M. Zentner, “Assessing musical abilities objectively: Construction and validation of the profile of music perception skills,” *PloS one*, vol. 7, no. 12, p. e52508, 2012.
- [17] S. Raptis, A. Chalamandaris, A. Baxevas, A. Askenfelt, E. Schoonderwaldt, K. F. Hansen, D. Fober, S. Letz, and Y. Orlarey, “Imutus-an effective practicing environment for music tuition,” in *Proc. Int. Computer Music Conf. (ICMC2005)*, 2005.
- [18] G. Tambouratzis, K. Perifanos, I. Voulgari, A. Askenfelt, S. Granqvist, K. F. Hansen, Y. Orlarey, D. Fober, and S. Letz, “Vemus: An integrated platform to support music tuition tasks,” in *Proc. IEEE Int. Conf. on Advanced Learning Technologies*, 2008, pp. 972–976.
- [19] E. Hein, “Music games in education,” in *Learning, Education and Games. Volume One: Curricular and Design Considerations*, K. Schrier, Ed. ETC Press, 2014, pp. 93–108.
- [20] A. Vidwans, S. Gururani, C.-W. Wu, V. Subramanian, R. V. Swaminathan, and A. Lerch, “Objective descriptors for the assessment of student music performances,” in *Proc. AES Int. Conf. on Semantic Audio*, Erlangen, 2017.
- [21] S. Holland, “Learning about harmony with harmony space: an overview,” in *Music education: An artificial intelligence approach*. Springer, 1994, pp. 24–40.
- [22] E. Chew and A. R. Francois, “Interactive multi-scale visualizations of tonal evolution in MuSA. RT Opus 2,” *Computers in Entertainment*, vol. 3, no. 4, pp. 1–16, 2005.
- [23] T. W. Hedges and A. P. McPherson, “3D gestural interaction with harmonic pitch space,” in *Proc. Int. Comp. Music Conf. and Sound and Music Comput. Conf. (ICMC-SMC’13)*, 2013, pp. 103–108.
- [24] D. Johnson, B. Manaris, and Y. Vassilandonakis, “Harmonic navigator: An innovative, gesture-driven user interface for exploring harmonic spaces in musical corpora,” in *Proc. Int. Conf. on Human-Computer Interaction*, 2014, pp. 58–68.
- [25] H. Riemann, *Harmony simplified, or the theory of the tonal functions of chords*. Augener Ltd., 1896.
- [26] A. Baratè, L. A. Ludovico, and D. Malchiodi, “Fostering computational thinking in primary school through a lego®-based music notation,” *Procedia Computer Science*, vol. 112, pp. 1334–1344, 2017.
- [27] V. Manzo, “Software-assisted harmonic function discrimination,” *J. of Music, Technology & Education*, vol. 7, no. 1, pp. 23–37, 2014.

# Learning to Generate Music with BachProp

**Florian Colombo**

School of Computer Science  
and School of Life Sciences  
École Polytechnique Fédérale  
de Lausanne, Switzerland  
florian.colombo@epfl.ch

**Johanni Brea**

School of Computer Science  
and School of Life Sciences  
École Polytechnique Fédérale  
de Lausanne, Switzerland  
johanni.brea@epfl.ch

**Wulfram Gerstner**

School of Computer Science  
and School of Life Sciences  
École Polytechnique Fédérale  
de Lausanne, Switzerland  
wulfram.gerstner@epfl.ch

## ABSTRACT

As deep learning advances, algorithms of music composition increase in performance. However, most of the successful models are designed for specific musical structures. Here, we present BachProp, an algorithmic composer that can generate music scores in many styles given sufficient training data. To adapt BachProp to a broad range of musical styles, we propose a novel representation of music and train a deep network to predict the note transition probabilities of a given music corpus. In this paper, new music scores generated by BachProp are compared with the original corpora as well as with different network architectures and other related models. A set of comparative measures is used to demonstrate that BachProp captures important features of the original datasets better than other models and invite the reader to a qualitative comparison on a large collection of generated songs.

## 1. INTRODUCTION

In search of the computational creativity frontier [1], machine learning algorithms are more and more present in creative domains such as painting [2, 3] and music [4–6]. Already in 1847, Ada Lovelace predicted the potential of analytical engines for algorithmic music composition [7]. Current models of music generation include rule based approaches, genetic algorithms, Markov models or more recently artificial neural networks [8].

One of the first artificial neural networks applied to music composition was a recurrent neural network trained to generate monophonic melodies [9]. In 2002, networks of long short-term memory (LSTM) [10] were applied for the first time to music composition, so as to generate Blues monophonic melodies constrained on chord progressions [11]. Since then, music composition algorithms employing LSTM units, have been used to generate monophonic [4, 5] and polyphonic music [6, 12–14] or to harmonize chorales in the style of Bach [6, 14]. However, most of these algorithms make strong assumptions about the structure of the music they model.

Here, we present a neural composer algorithm named *BachProp* designed to generate new music scores in an arbitrary style implicitly defined by the corpus of training data. To this end, we do not assume any specific musical structure of the data except that it is composed of sequences of notes that are characterized by pitch, duration and time-shift relative to the previous note.

In the following, we start by contrasting our representation of music to previous propositions [6, 12, 14, 15] with a focus towards training style-agnostic generative models of music. We then introduce our algorithm and compare BachProp with other models on a standard datasets of chorales written by Johann Sebastian Bach [16] and establish new benchmarks on the musically complex datasets of MIDI recordings by John Sankey [17] and string quartets by Haydn and Mozart [18]. Finally, as the evaluation and comparison of generative models is not trivial [19], we invite the reader, first, to a subjective comparison on a large collection of samples generated from the different models on the accompanying media webpage [20] and, second, we propose a new set of metrics to quantify differences between the models. Preliminary versions of our work have been made available on arXiv [21, 22].

## 2. RELATED WORK

Unlike approaches to image generation, where the standard data consists of rows and columns of pixel values for multiple color channels, approaches to music generation lack a standard representation of music data. This is reflected by the zoo of music notation file formats (ABC, LilyPond, MusicXML, NIFF, MIDI) and the fact that lossless conversion from one to the other is usually not possible. The MIDI file format captures most features of music, like polyphony, dynamics, micro tuning, expressive timing and tempo changes. But its representational richness and the possibility to represent the exact same song in multiple ways, make it challenging to work directly with MIDI. Therefore, all approaches discussed in the following use a first preprocessing step to transform all songs into a simpler representation. The subsequent design choices of the generative model are heavily influenced by this first preprocessing step.

DeepBach [6] is designed exclusively for songs with a constant number of voices (e.g. four voices for a typical Bach chorale) and a discretization of the rhythm into multiples of a base unit, e.g. 16<sup>th</sup> notes. The model achieves

Copyright: © 2019 Florian Colombo et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

good results not only in generating novel songs but allows also in reharmonizing given melodies while respecting user-provided meta-information like the temporal position of fermatas. The model works with a Gibbs-sampling-like procedure, where, for each voice and time step, one note is sampled from conditional distributions parameterized by deep neural networks. The conditioning is on the other voices in a time window surrounding the current time-step. Additionally a “temporal backbone” signals the position of the current 16<sup>th</sup> note relative to quarter notes and other meta-information. A special hold symbol can also be sampled instead of a note, to represent notes with a duration longer than one time-step.

BachBot [14] and its Magenta implementation Polyphony-RNN [15] contain no assumption about the number of voices; they can be fit to any corpus of polyphonic music, if the rhythm can be discretized into multiples of a base unit, e.g. 16<sup>th</sup> notes. Songs are represented as sequences of NEW\_NOTE(PITCH), CONT\_NOTE(PITCH) and STEP\_END events, where the STEP\_END event indicates the end of the current time-step. Between two STEP\_END events, typically several NEW\_NOTE(PITCH) and CONT\_NOTE(PITCH) events can be found sorted by PITCH. A generative model parametrized by a recurrent neural network model is fit to these sequences of events, in the same way as recurrent neural network models are used for language modeling on a character- or word-level [23–25].

Common to the models discussed above is a discretization of time into multiples of a base unit like the 16<sup>th</sup> note. This limits the representable rhythms considerably; e.g. triplets, grace notes or expressive variations in timing cannot be represented in this way. To overcome this limitation, [26] replace the repertoire of symbols employed by the Polyphony-RNN by NOTE\_ON, NOTE\_OFF, TIME\_SHIFT and SET\_VELOCITY events, where the TIME\_SHIFT events allows the model to move forward in time by multiples of 8 ms up to 1 second and the SET\_VELOCITY events allow to model the loudness of a note (which depends on the piano on the velocity with which a key is pressed).

### 3. METHOD

In written music, the  $n^{\text{th}}$  note  $\text{note}[n]$  of a piece of music  $\text{song} = (\text{note}[1], \dots, \text{note}[N])$  can be characterized by its pitch  $P[n]$ , duration  $T[n]$  and the time-shift  $dT[n]$  of its onset relative to the previous note, i.e.  $\text{note}[n] = (dT[n], T[n], P[n])$ . The time-shift  $dT[n]$  is zero for notes played at the same time as the previous note. In contrast to most other approaches that discretize the time into multiples of a base unit (except e.g. [26]), we round all durations into a set of defined musical durations which allows a more faithful representation of timing that is limited only by the number of possible values considered for  $T[n]$  and  $dT[n]$ . For example, our representation allows to easily and without any distortion represent 32<sup>nd</sup> notes, triplets and double dotted notes in the same dataset. As well as any other more complex note durations that can be needed for specific corpora. To achieve that, each duration in the original corpus is mapped to the closest duration in a set of integer multiples of atomic durations. For this work, we considered 64<sup>th</sup>

and 32<sup>nd</sup> triplets as atomic durations.

Our approach is to approximate probability distributions over note sequences in music scores  $\text{song}_1, \dots, \text{song}_S$  with distributions parameterized by recurrent neural networks and move its weights  $\theta$  towards the maximum likelihood estimate

$$\theta^* = \arg \max_{\theta} Pr(\text{song}_1, \dots, \text{song}_S | \theta), \quad (1)$$

Since each note in each song consists of the triplet  $(dT, T, P)$  we can parametrize the distributions in a similar way as the pixel-RNN [27] that was developed for the (red, green, blue) triplets of pixels in images. Importantly, our model takes into account that pitch and duration of a note are generally not independent. For example in classical music, the fundamental, e.g. the note C in a piece written in C major, tends to be longer than other notes.

In the following we describe in more details our representation of music, the structure of the model and our approach to comparing different models that use different representations of music.

#### 3.1 Conversion of MIDI files into our representation of music

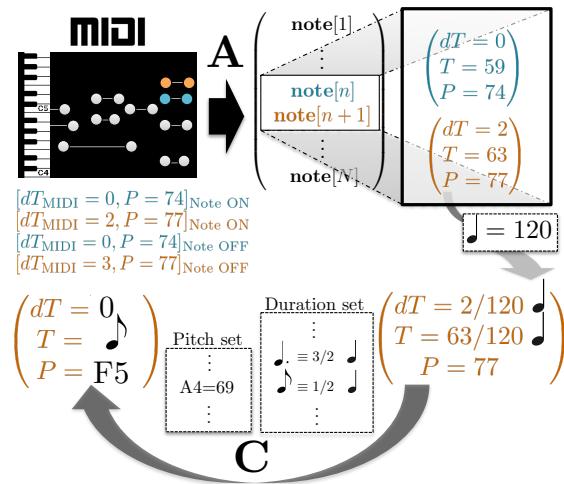


Figure 1. **From MIDI to our representation of music.** An illustration of the steps involved in the proposed conversion of MIDI sequences. See text for details.

A MIDI file contains a header (meta parameters) and possibly multiple tracks that contain a sequence of MIDI messages. For BachProp, we merge all tracks and consider only the MIDI messages defining when a note starts (ON events) or ends (OFF events). For each ON event we look forward at the next OFF event with the same pitch  $P$  to convert sequences of MIDI messages into a sequences of notes (Figure 1A). We then translate timings from the internal MIDI TICK representation to quarter note lengths (Figure 1B).

Next, we round all durations  $T[n]$  such that they are in the set of possible note lengths (duration set in Figure 1C) expressed in units of a quarter note, similar to durations in standard music notation software. Similarly, we round the



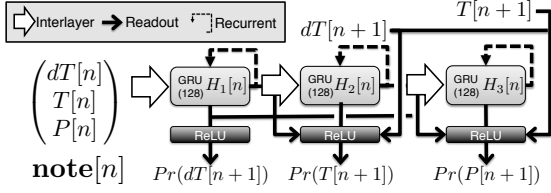


Figure 2. **BachProp neural architecture.** See text for details.

time-shifts  $dT[n]$  to 0 or one of the possible note lengths. Mapping to the closest value in the set removes temporal jitter around the standard note duration that may have been introduced accidentally at the moment of recording the MIDI file (Figure 1C). While this standardization may be desired when expressive timing is not taken into account, it is straightforward to extend the duration dictionary to include also values that allow to model expressive timing. However, we believe that a good generative model of music should be trained to model the music structure only. If the music representation is introducing additional temporal dependencies, the model will need to learn these interfering structures as well. In that sense, the processing we designed for BachProp and presented in this section allows it to focus on the essential structure of music.

In order for BachProp to learn tonality, during training and before each new epoch, we randomly transpose every song within the available bounds of the pitch set. For each song we compute one of the possible shift of semitones and apply it as an offset to all pitches within the song. Because a single MIDI sequence will be transposed with up to 20 offsets, this augmentation method allows BachProp to learn the temporal structure of music on more examples.

Finally, we add an artificial note at the beginning and end of each score. After training, the inaudible ‘boundary note’ is used by the model to seed and end the generation of songs.

### 3.2 The BachProp neural network

We employ a deep GRU [28] network with three consecutive layers as schematized in Figure 2. The network’s task is to infer the probability distribution over the next possible notes from the representation of the current note and the network’s internal state (the network representation of the history of notes).

The probability of a sequence of  $N$  notes  $\text{note}[1 : N] = (\text{note}[1], \dots, \text{note}[N])$  is given by

$$Pr(\text{note}[1 : N]) = Pr(\text{note}[1]) \prod_{n=1}^{N-1} Pr(\text{note}[n+1] | \text{note}[1 : n]). \quad (2)$$

Each term on the right hand side can be further split into

$$\begin{aligned} Pr(\text{note}[n+1] | \text{note}[1 : n]) = & Pr(dT[n+1] | \text{note}[1 : n]) \times \\ & Pr(T[n+1] | \text{note}[1 : n], dT[n+1]) \times \\ & Pr(P[n+1] | \text{note}[1 : n], dT[n+1], T[n+1]). \end{aligned} \quad (3)$$

The goal of training the Bachprop network with parameters  $\theta$  is to approximate the conditional probability distributions on the right hand side of Equation (3).

In the BachProp network (Figure 2), the conditioning on the history  $\text{note}[1 : n]$  is implemented by the values of the shared hidden states. The hidden state is composed of 3 recurrent layers with 128 gated-recurrent units (GRU). The state  $H_1[n]$  of the first hidden layer is updated with input  $\text{note}[n]$  and previous state  $H_1[n-1]$ . The state of the upper layers  $H_i[n]$  for  $i = 2, 3$  are updated with inputs  $H_{i-1}[n]$  and  $H_i[n-1]$ .

$\text{note}[n]$  is represented by three one-hot vectors encoding separately  $dT[n]$ ,  $T[n]$  and  $P[n]$ , i.e. every entry in these vectors is 0 but the one mapped to the value  $x[n]$  for  $x = dT, T, P$ . The length of each of these vectors is defined by the size of the respective dictionary  $L_x$ . Therefore, each song is encoded as a set of three 2-dimensional binary matrices of size  $N_s \times L_x$ , where  $N_s$  stands for number of notes in song  $s$  and  $L_x$  for the size of the set of unique time-shifts ( $x = dT$ ), note durations ( $x = T$ ) or keys ( $x = P$ ) present in the original corpus. These matrices are zero-padded to account for variable input lengths  $N_s$ .

During training, songs are presented to the network in sequences of batches containing binary tensors with dimension  $B \times N \times L$ .  $B = 32$  is the number of songs presented together over which the gradient of the error signal is averaged.  $N = 128$  is the number of consecutive notes over which the gradients are computed, i.e. we used truncated backpropagation through time.  $L = L_{dT} + L_T + L_P$  is the size of the entire input vector encoding for  $\text{note}[n]$ . Though the gradient is truncated, the recurrent units are stateful: they maintain their states across consecutive batches while the same sets of  $B$  songs are being presented. When a new batch of  $B$  songs is being presented to the network, hidden states are reset.

To generate  $\text{note}[n+1]$ , one third ( $H_1[n]$  in Figure 2) of the full hidden state is fed into a feedforward network with one layer of Rectified linear (ReLU) units and one output softmax layer that represents  $Pr(dT[n+1] | H_1[n]) \approx Pr(dT[n+1] | \text{note}[1 : n])$ . The chosen  $dT[n+1]$  together with  $H_1[n]$  and  $H_2[n]$  is fed into a second feedforward network with one layer of ReLU units and an output softmax layer that represents  $Pr(T[n+1] | H_1[n], H_2[n], dT[n+1]) \approx Pr(T[n+1] | \text{note}[1 : n], dT[n+1])$ . In a similarly way, the pitch is sampled from  $Pr(P[n+1] | H_1[n], H_2[n], H_3[n], dT[n+1], T[n+1]) \approx Pr(P[n+1] | \text{note}[1 : n], dT[n+1], T[n+1])$ . The ReLU and softmax readout layers have the same size  $L_x$  as the dictionary of the feature they model. These three small steps of sampling  $dT[n+1]$ ,  $T[n+1]$  and  $P[n+1]$  form together one big step from note  $n$  to note  $n+1$ .

The resulting sequence of notes is a newly generated score sampled from BachProp. Note that, the temperature of sampling can be adapted to the confidence we give to the model predictions [5, 29]. In particular, any model trained with a corpus that exhibits many repetition of patterns, will generate scores with more examples of these repetitions for lower sampling temperatures. Indeed, a lower temperature

will reduce the probability to select an undesired note that is not part of the pattern to be repeated. Finally, the generated sequence of notes in our representation can easily be translated back to a MIDI sequence by reversing the method schematized in Figure 1.

BachProp has been implemented in Python using the Keras API [30]. Code is available on GitHub<sup>1</sup>.

### 3.3 Comparison against plagiarism and other models

Even in well-established domains such as computer vision and image generation, it is not clear how to compare generative models [19]. But in order to turn generative models of music eventually into useful tools for composers, they should be able to generate (1) plagiarism-free music of (2) a predefined style or mood that is (3) pleasant to listen to.

A way of measuring plagiarism is to control overfitting by comparing the loss on training and validation data. While this is a simple method it is rather coarse since it works on songs as a whole. Instead we propose *novelty profiles* that compare the co-occurrence of short note sequences across different data sets. A crucial parameter of novelty profiles is the length of a note sequence on which the comparison takes place. We adapted the novelty profile, a measure of similarity between any given score and a reference corpus, from [5]. For a pattern size of 6 notes, a novelty score of 1 indicates that all patterns of 6 consecutive notes are not present in the reference corpus. On the other hand, a note sequence that contains only patterns found in the reference corpus would exhibit a novelty score of 0. We define the binary novelty of a single pattern by checking if all three features ( $dT[n-m:n]$ ,  $T[n-m:n]$ ,  $P[n-m:n]$ ) of the notes included in the pattern are found in the same order anywhere in the reference corpus. The novelty score of an entire song is the average binary novelty over all possible patterns.

Models that are trained on the same representation of music can be compared by their likelihood to assess how well they generate pieces of a predefined type. But if the models represent probability distributions over different spaces, which is quickly the case when different representations are used, they are unfortunately not comparable in terms of likelihood. For example, the event based representation from [26] can in principle produce all possible note sequences. But it could also generate nonsensical sequences of multiple consecutive NOTE\_OFF events, without corresponding previous NOTE\_ON events. To nevertheless compare models that build on different representations of music we propose simple statistics like interval distributions that can be applied to the samples of each generative model of music.

Finally, to compare the pleasantness of the generated music, one can ask people to rate different pieces; an approach that is followed in previous works (e.g. [6]). We also invite the reader to listen to the large collections of non-cherry-picked generated examples [20].

## 4. RESULTS AND DISCUSSION

### 4.1 Datasets

We consider four MIDI corpora with different musical structures and styles (see Table 2). The Nottingham database [31] contains British and American folk tunes. The musical structure of all songs is very similar with a melody on top of simple chords. The Chorales corpus [16] includes hundreds of four-part chorales harmonized by Bach. All chorales share some common structures, such as the number of voices and rhythmical patterns. For comparison we used the same filtering of songs as DeepBach [32] to exclude chorales with number of voices unequal four. We consider both Nottingham and Chorales corpora as homogeneous data sets. The John Sankey data set [17] is a collection of MIDI sequences recorded by John Sankey on a digital keyboard. Even though all songs were composed by Bach, the pieces are rather different. In addition, this data set was recorded live from the digital keyboard and thus we applied the temporal normalization described above. At last, the string quartets data set [18] includes string quartets from Haydn and Mozart. Here again, there is a large heterogeneity of pieces across the corpus.

Renderings of original and generated scores are available for listening on the webpage containing media for this paper<sup>2</sup> [20]. To train BachProp on the different corpora, we used the same network architecture, number of neurons, initialization and learning parameters, but each of the network was trained on a different corpus.

### 4.2 Alternative models

In addition to BachProp, we trained six other models; three BachProp variants to assess the impact of our design choices, one baseline model and two previously published and available artificial composers. PolyDAC and IndepBP are direct BachProp variants. MidiBP is a version of BachProp that utilizes a different representation of MIDI note sequences inspired by [26]. The two state-of-the-art artificial composers, DeepBach [6] and PolyRNN [15] allow us to compare scores generated by models of our design with other algorithms. The 6th model is a multi-layer perceptron model (MLP) and serves as a baseline control.

**PolyDAC** is a polyphonic version of [5]. It models the same conditional distribution as BachProp but instead of reading out the probabilities from shared hidden layer states, it models each note feature with three independent neural networks. The time-shift, duration, and pitch networks are composed of three recurrent layers with 16, 128, and 256 GRUs respectively. **IndepBP** assumes that all note features are independent from each others. As such,  $Pr(dT[n+1])$ ,  $Pr(T[n+1])$ , and  $Pr(P[n+1])$  are read out by three softmax output layers directly from the hidden state of three hidden layers composed of 128 GRUs that takes as input the one-hot encoding of the  $n^{th}$  note. **MidiBP** neural architecture consists of three recurrent layers composed of 128 GRUs. Here, the MIDI note sequences are represented differently. While the normalization and pre-processing is done as described above (Figure 1), we then

<sup>1</sup> <https://github.com/FlorianColombo/BachProp>

<sup>2</sup> Media webpage: <https://goo.gl/Z4AfPg>

convert the normalized music score back to the MIDI-like format proposed in [26] where in each time step a single on-hot vector defines either a NOTE\_ON event and its corresponding pitch, a NOTE\_OFF event and its corresponding pitch, or a time-shift and its corresponding duration (defined by our duration representation). Therefore, a single softmax read out layer is used to sample the upcoming MIDI event. **MLP** has no recurrent layers but 3 feedforward hidden layers of 124 ReLUs each that gets as input the 5 most recent notes  $\text{note}[n-4:n]$  together with the current time-shift  $dT[n+1]$  and duration  $T[n+1]$  to sample the pitch  $P[n+1]$ . To sample the duration  $T[n+1]$  and the time-shift  $dT[n+1]$ , appropriate parts of the input are masked with zeros.

Models BachProp, PolyDAC, MidiBP, IndepBP were trained with truncated back propagation through time and the Adam optimizer [33]. The MLP model was trained with standard back propagation and the Adam optimizer. The mini-batch size is 32 scores, the validation set a 0.1 fraction of the original corpus, and one training epoch consists of updating the network parameters with all training examples and evaluating the performances on the entire validation set. Training is stopped when the performances on the validation set saturates and the model leading to the highest accuracy is used for generating new music scores. DeepBach was trained for 15 epochs with the standard settings of the current master branch [32]. PolyRNN was trained for 26000 steps with the standard settings of the current master branch [15].

MODEL	NLL	$dT$	$T$	$P$
BACHPROP	0.419	0.97	0.91	0.77
POLYDAC	0.647	0.97	0.94	0.69
INDEPBP	0.647	0.97	0.75	0.63
MLP	0.796	0.95	0.76	0.49

Table 1. **Comparison of architectures on our representation of music.** NLL stands for negative log-likelihood on the validation set. Columns  $dT$ ,  $T$  and  $P$  indicate the accuracy (fraction of correct predictions) for time-shifts, durations and pitches, respectively.

### 4.3 BachProp performs better than alternative models with same representation

On the Bach Chorales we find that the BachProp architecture performs considerably better than the alternative architectures using the same representation of music (see Table 1). As expected, the standard feedforward MLP with ReLUs yields the worst performance. It lacks the ability to model long range dependencies, which the other models can do through their recurrent connections. When we remove the conditioning on each of probability terms on the right side of Equation (3), as done for the IndepBP model, we get poorer performances. We further observe that sharing a common hidden state allowed BachProp to outperform PolyDAC on the pitch predictions.

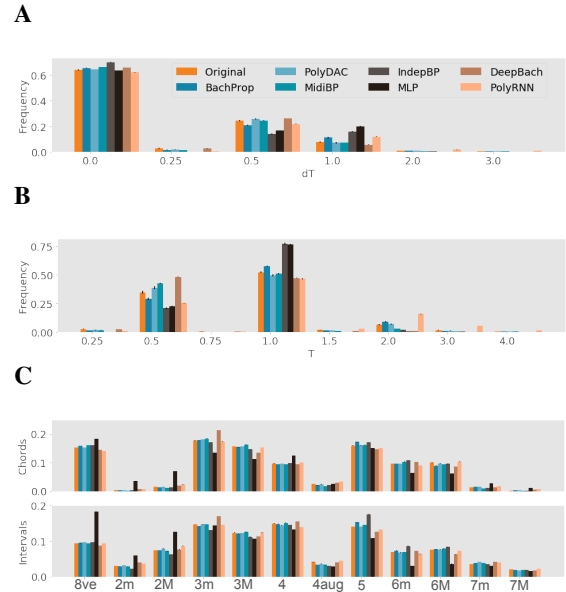


Figure 3. **Local statistics.** **A** Distribution of  $dT$ . **B** Distribution of  $T$ . **C** Distribution of intervals in chords (top) and between each note (bottom). For all figures, we show the mean and standard deviation (in black) obtained with bootstrapping (50% of the entire corpus resampled 10 times). All models were trained on the Bach Chorales corpus.

### 4.4 BachProp performs at least as good as alternatives with different representation

To compare models that use a different representation of music, we look at a set of metrics that includes local statistics, song-length statistics and novelty profiles. To evaluate these metrics for each model, we generated from each model a set containing as many scores as the original Bach Chorales corpus. We include the baseline models from the last section for comparison reasons.

#### 4.4.1 Local statistics

A model that has captured the underlying structure of the sequences of notes present in a corpus, should be able to generate new scores matching the local statistics of what they modeled. As such, we suggest to compute the distributions of generated  $dT$  and  $T$  and compare them to the original corpus distributions as a first metric to evaluate generative models of music. Note that for such direct local statistics, a simple n-gram model would match the original distributions perfectly. Figure 3A and B shows that BachProp and PolyDAC match the original distributions best, followed by MidiBP, DeepBach and PolyRNN, while IndepBP and MLP match the least.

Next, we look at interval distributions. An interval is the number of half-tone separating two notes. Here, BachProp, PolyDAC, MidiBP and PolyRNN match the distribution quite well. DeepBach seems to generate minor thirds considerably more often than present in the training data (Figure 3C).

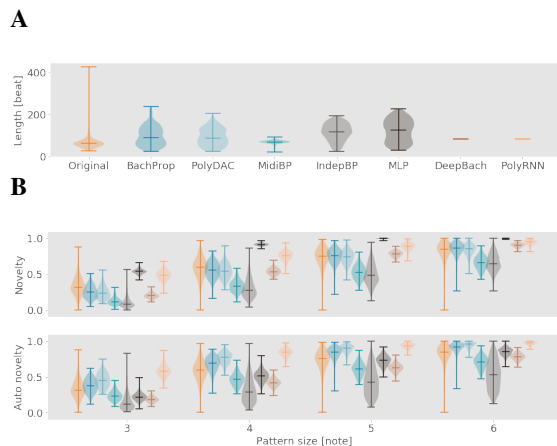


Figure 4. **Song lengths and novelty profiles.** **A** Distribution of the duration of scores in quarter note length. **B** Novelty profile of all corpora with respect to the auto-novelty of the original corpus (top). The auto-novelty profiles of all corpora (bottom). See text for details.

#### 4.4.2 Distribution of song lengths

The distribution of song lengths can indicate whether a model captured really long-range dependencies in the training set. On this measure MidiBP matches the distribution slightly better than BachProp, PolyDAC, IndepBP and MLP (see Figure 4A). Since DeepBach and PolyRNN do not model score endings, we manually set their duration.

#### 4.4.3 Novelty profiles

In Figure 4B (top), we compare the novelty profiles for all models with respect to the original Choraes corpus with which each model was trained. We compare the different profiles with the auto-novelty of the reference corpus. The auto-novelty is the novelty profile for each song in the reference corpus with respect to the same corpus without the song for which the novelty score is computed. It reflects, how similar is the music within the original corpus and is consequently the distribution to match for an ideal generative model of music. Here, the only model that is clearly outside the target distribution is the MLP model. While the IndepBP and MidiBP models match the target distributions, their novelty distributions for bigger pattern sizes is lower than the original corpus auto-novelty. This is an indicator that these models are generating music examples that are too similar to the original data. In other words, these models adopted a strategy closer to reproducing or recombining observed patterns rather than inferring the actual temporal dependencies between music notes. DeepBach, BachProp and PolyDAC have their medians close and above the original distributions. However, DeepBach and PolyRNN have a surprisingly low variance for each of the pattern sizes.

In Figure 4B (bottom) we compare the auto-novelty of all generated corpora with the original corpus. An auto-novelty profile exhibiting distributions with lower novelty scores than the original data set, is suspected to generate new music scores of little diversity. The auto-novelty profile of BachProp and PolyDAC match the one of the origi-

DATASET	NLL	$dT$	$T$	$P$	SIZE [SCORE — NOTE]
CHORALES	0.419	0.97	0.91	0.77	357 — 95'337
NOTTINGHAM	0.587	0.98	0.89	0.70	1037 — 313'975
JOHN SANKEY	1.002	0.89	0.77	0.45	135 — 358'211
STRING QUARTETS	0.936	0.88	0.83	0.49	215 — 738'739

Table 2. **BachProp on other datasets.** See Table 1 for description of labels.

nal corpus best.

#### 4.5 BachProp generates pleasant examples on more complex datasets

As a reference for future comparisons, we report here the results of BachProp trained on more complex datasets. In Table 2, we observe that for homogeneous corpora with many examples of similar structures (Choraes, Nottingham), BachProp can predict notes with higher accuracies than for more heterogeneous data sets (John Sankey, String Quartets).

We encourage readers to listen to the examples provided on the accompanying webpage [20] to convince themselves of the ability of BachProp and its variants to generate unique and heterogeneous new music scores.

## 5. CONCLUSION

In this paper, we presented BachProp, an algorithm for general automated music composition. Our main contributions are (1) a note-sequence based representation of music with minimal distortion of the rhythm for training neural network models, (2) a network architecture that learns to generate pleasant music in this representation and (3) a set of metrics to compare generative models that operate on different representations of music.

BachProp can be used both for automated and interactive music composition. Indeed, adding a human composer in the composition process is straightforward and BachProp can then be used as a computer aided music composition algorithm. For example, the human composer could use BachProp to suggest possible continuations and select among them. With such algorithms, the authors foresee that music composition can be brought to a wider audience, therefore allowing untrained humans to compose their own pieces of music.

## Acknowledgments

This research was partially supported by the Swiss National Science Foundation (Grant 200020\_165538) and the Laboratory of Computational Neuroscience (EPFL-LCN).

## 6. REFERENCES

- [1] S. Colton, G. A. Wiggins *et al.*, “Computational creativity: The final frontier?” in *ECAI*, vol. 12, 2012, pp. 21–26.
- [2] A. Mordvintsev, C. Olah, and M. Tyka, “Inceptionism: Going deeper into neural networks,” *Google Research Blog*. Retrieved June, vol. 20, no. 14, p. 5, 2015.



- [3] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2016, pp. 2414–2423.
- [4] B. L. Sturm, J. F. Santos, O. Ben-Tal, and I. Korshunova, “Music transcription modelling and composition using deep learning,” in *1st Conference on Computer Simulation of Musical Creativity*, 2016.
- [5] F. Colombo, A. Seeholzer, and W. Gerstner, “Deep artificial composer: A creative neural network model for automated melody generation,” in *International Conference on Evolutionary and Biologically Inspired Music and Art*. Springer, 2017, pp. 81–96.
- [6] G. Hadjeres, F. Pachet, and F. Nielsen, “DeepBach: a steerable model for Bach chorales generation,” in *34th International Conference on Machine Learning*, vol. 70. JMLR, 2017, pp. 1362–1371.
- [7] A. Lovelace, “Notes on L. Menabrea’s ‘sketch of the analytical engine invented by Charles Babbage, esq.’,” *Taylor’s Scientific Memoirs*, 1843.
- [8] J. D. Fernández and F. Vico, “AI methods in algorithmic composition: A comprehensive survey,” *Journal of Artificial Intelligence Research*, vol. 48, pp. 513–582, 2013.
- [9] P. M. Todd, “A connectionist approach to algorithmic composition,” *Computer Music Journal*, vol. 13, no. 4, pp. 27–43, 1989.
- [10] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [11] D. Eck and J. Schmidhuber, “Finding temporal structure in music: Blues improvisation with LSTM recurrent networks,” in *12th IEEE Workshop on Neural Networks for Signal Processing*. IEEE, 2002, pp. 747–756.
- [12] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, “Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription,” in *29th International Conference on Machine Learning*, 2012.
- [13] S. Lattner, M. Grachten, and G. Widmer, “Imposing higher-level structure in polyphonic music generation using convolutional restricted Boltzmann machines and constraints,” *Journal of Creative Music Systems*, vol. 2, no. 1, 2018.
- [14] F. Liang, M. Gotham, M. Johnson, and J. Shotton, “Automatic stylistic composition of Bach chorales with deep LSTM,” in *18th International Society for Music Information Retrieval Conference*, 2017.
- [15] Magenta Team Google Brain, “Polyphony RNN, revision ca73164,” [https://github.com/tensorflow/magenta/tree/master/magenta/models/polyphony\\_rnn](https://github.com/tensorflow/magenta/tree/master/magenta/models/polyphony_rnn), 2016.
- [16] “J.S. Bach chorales,” <http://web.mit.edu/music21/>, accessed: 2019-04-01.
- [17] “Bach MIDI sequences by John Sankey,” <http://www.jsbach.net/midi/midi-johnsankey.html>, accessed: 2019-04-01.
- [18] “String quartets by Mozart and Haydn,” <http://www.stringquartets.org>, accessed: 2019-04-01.
- [19] L. Theis, A. van den Oord, and M. Bethge, “A note on the evaluation of generative models,” *ArXiv:1511.01844*, p. arXiv:1511.01844, 2015.
- [20] “BachProp media webpage,” <https://sites.google.com/view/bachprop>, accessed: 2019-04-01.
- [21] F. Colombo and W. Gerstner, “BachProp: Learning to compose music in multiple styles,” *arXiv preprint arXiv:1802.05162*, 2018.
- [22] F. Colombo, J. Brea, and W. Gerstner, “Learning to generate music with BachProp,” *arXiv preprint arXiv:1812.06669*, 2018.
- [23] I. Sutskever, J. Martens, and G. Hinton, “Generating text with recurrent neural networks,” in *28th International Conference on Machine Learning*, 2011, pp. 1017–1024.
- [24] A. Graves, “Generating Sequences With Recurrent Neural Networks,” *ArXiv:1308.0850*, 2013.
- [25] T. Mikolov, “Statistical language models based on neural networks,” Ph.D. dissertation, 2012.
- [26] S. Oore, I. Simon, S. Dieleman, and D. Eck, “Learning to create piano performances,” *NIPS 2017 Workshop on Machine Learning for Creativity and Design*, 2017.
- [27] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, “Pixel Recurrent Neural Networks,” *ArXiv:1601.06759*, 2016.
- [28] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.
- [29] A. Karpathy, “The unreasonable effectiveness of recurrent neural networks,” <http://karpathy.github.io/2015/05/21/rnn-effectiveness>, 2016.
- [30] F. Chollet *et al.*, “Keras,” <https://github.com/fchollet/keras>, 2015.
- [31] “Nottingham data set of folk songs,” <http://www-etud.iro.umontreal.ca/~boulanni/icml2012>, accessed: 2019-04-01.
- [32] “DeepBach, revision f069695,” <https://github.com/Ghadjeres/DeepBach>, accessed: 2019-04-01.
- [33] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

# OFFLINE SCORE ALIGNMENT FOR REALISTIC MUSIC PRACTICE

Yucong Jiang<sup>1</sup>    Fiona Ryan<sup>1,2</sup>    David Cartledge<sup>2</sup>    Christopher Raphael<sup>1</sup>

<sup>1</sup> School of Informatics, Computing, and Engineering    <sup>2</sup> Jacobs School of Music

Indiana University Bloomington, USA

yujiang, fkryan@iu.edu    docartle, craphael@indiana.edu

## ABSTRACT

In a common music practice scenario a player works with a musical score, but may jump arbitrarily from one passage to another in order to drill on difficult technical challenges or pursue some other agenda requiring non-linear movement through the score. In this work we treat the associated score alignment problem in which we seek to align a known symbolic score to audio of the musician's practice session, identifying all "do-overs" and jumps. The result of this effort facilitates a quantitative view of a practice session, allowing feedback on coverage, tempo, tuning, rhythm, and other aspects of practice. If computationally feasible we would prefer a globally optimal dynamic programming search strategy; however, we find such schemes only barely computationally feasible in the cases we investigate. Therefore, we develop a computationally efficient off-line algorithm suitable for practical application. We present examples analyzing unsupervised and unscripted practice sessions on clarinet, piano and viola, providing numerical evaluation of our score-alignment results on hand-labeled ground-truth audio data, as well as more subjective and easy-to-interpret visualizations of the results.

## 1. INTRODUCTION

### 1.1 Problem Description

Score alignment finds a correspondence between a symbolic representation of a musical score and an associated audio performance, identifying the positions of all note onsets. The subject was introduced through the early musical accompaniment systems of Dannenberg and Vercoe [1,2], while notable contributions include [3–13]. Cuvillier [3] provides a thorough review on score alignment. This paper deals with a variation of the traditional score alignment problem: instead of aligning a performance, we align the audio of a music practice session of a given score, where the player is allowed to skip from one score location to another in an arbitrary fashion. Such a score-alignment problem is also called score-alignment-with-skips, dropping the constraint of linear movement in the score while playing.

Traditional score alignment is typically partitioned into two varieties: on-line and off-line. On-line recognition

is appropriate for applications in which the audio source must be understood in real-time, such as musical accompaniment systems, automatic page-turners, or approaches that coordinate the display of subtitles for opera. Off-line recognition is appropriate for score alignment applications without a real-time component, such as note-level editing of an audio performance, quantitative analysis of musical performance from a stylistic viewpoint, or the automatic generation of large sample libraries from recordings.

In this effort, we treat the *off-line* recognition of a music practice session. In particular, we target typical instrumental practice, which usually involves a large degree of repetition with particular attention directed toward challenging passages, as well as frequently inaccurate playing. The score-alignment-with-skips problem can enable a number of meaningful applications, as follows.

### 1.2 Applications

Off-line score-alignment-with-skips allows the development of useful tools that provide a high-level view of a practice session. Our experiments present a fledgling version of such a tool, facilitating efficient navigation through a practice session while coordinating the audio and visual display of the score. We believe the understanding provided by such a tool is far superior to that gained by simply listening to a practice recording, and is particularly important to musicians working to develop effective practice techniques. Interacting with such a tool implicitly answers a variety of useful questions, such as how much time was spent on a particular passage, or what was the typical length of a repeated fragment. The underlying analysis also enables additional feedback about tuning, tempo and rhythm, providing a deeper level of pedagogical feedback. For example, most wind players will have particular notes that are consistently flat or sharp, while such *global* tuning characteristics could easily be computed from the results of our proposed score alignment. Another example would be identifying a common problem of the student learner — unconsciously reducing the tempo when technical difficulties are encountered. Perhaps one could even develop measures of improvement over the course of a sequence of practice sessions.

Tools that facilitate navigating a practice session efficiently, perhaps including useful summaries of the session, offer particularly engaging possibilities for the music teacher as well. Many teachers experience the interval between lessons as something of a "black box," where divining the difference between a student's self-perceptions and real-

Copyright: © 2019 Yucong Jiang et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

ity can be a challenge. Currently, the closest we can get to practice intervention is to directly observe practice, or to have students submit recordings of their practice, both of which are linear, and in real time. If a teacher were able to view a high-level representation of practice over the course of a week, he or she could intervene and correct the students at a new and productive level of granularity, with reference to a concrete analysis of practice over time.

The score-alignment-with-skips variant has an on-line version as well, untreated here, though discussed in Nakamura et al. [6]. Such technology would be appropriate for a system that interacts with a musician during practice. For instance, after having identified the section that is currently being rehearsed, a system may provide an accompaniment that follows and supports the player. Fertile possibilities also exist for musical tutoring systems. For instance, at any time during a practice session we may add a metronome whose rate and phase initially synchronize with the live player, proceeding either deterministically or in an adaptive manner. We may also periodically suggest interventions over the course of the session, such as slow practice, or directing the subject's attention toward unmet challenges.

### 1.3 Related Work

One version of the score-alignment-with-skips problem is treated by Müller and Appelt [13], where the authors seek to compare different versions of a piece of music, perhaps with different choices of repeats, though with a preference for matching long sections of the two audio recordings, unlike what would be encountered with instrumental practice.

More recently, Nakamura [6] treats a version more oriented to our vision of practice analysis. This method performs online analysis by computing the filtered distribution without approximation through the usual “forward” iteration. With this approach, as with globally optimal dynamic programming computation of the most likely path, the computation is  $O(NS)$ , where  $N$  is the length of the data and  $S$  is the number of notes in the score. Nakamura observes that their algorithm is feasible in real time; however, from the computational complexity one can see that this depends on the particular score chosen. We imagine practice scenarios where the “score” might be a concatenation of all the scores in a player's library, thus nearly ruling out globally optimal approaches with no approximation. Furthermore, we expect that a more fine-grained approach will increase the number of states that must be devoted to each score note. For these reasons we pursue approaches that relax the guarantee of global optimality in exchange for both computational efficiency and extensibility to more complex graph topologies.

In our experiments we don't see a way to make direct comparisons with these approaches as Nakamura's work is on the filtering problem, thus not appropriate the off-line score alignment problem, while Müller considers a version of the problem that is far more constrained, thus not workable for kind of unconstrained practice considered in our experiments.

In what follows we explicitly describe our score align-

ment methodology. In addition, we present results on about two hours' worth of audio data on clarinet, viola and piano (polyphonic), collected from various members of the Jacobs School of Music at Indiana University, both in numerical fashion and through our audio-visual “practice browser.”

## 2. MODELING

In this section, we first describe a hidden Markov model for the score alignment problem, which serves as the foundation of our approach to the score-alignment-with-skips problem. Then, we explain the motivations behind our approach. Lastly, we explain using a pitch tree and beam search to accommodate the computation burden.

### 2.1 HMM for Score Alignment

Score alignment has been cast as a hidden Markov model (HMM) problem by several authors [3, 5, 6, 9, 12]. Here, we use the framework in [9].

As the HMM views time as discrete, we model time as a sequence of “frames” of about 30 ms. in length. We denote the hidden Markov chain as  $X = X_1, \dots, X_N$  where  $N$  is the number of frames in the audio excerpt, and  $X_n$  is the hidden state associated with the  $n$ th frame, taking values in a state graph. A simple construction of the state graph models the  $k$ th note as a chain of states,  $s_{k,1}, \dots, s_{k,M}$ , where  $M$  is the maximum length of the  $k$ th note, in frames. Figure 1 shows a topology where each state,  $s_{k,m}$ , either connects to  $s_{k,m+1}$ , the next state of the same note, or to  $s_{k+1,1}$ , the first state of the next note.

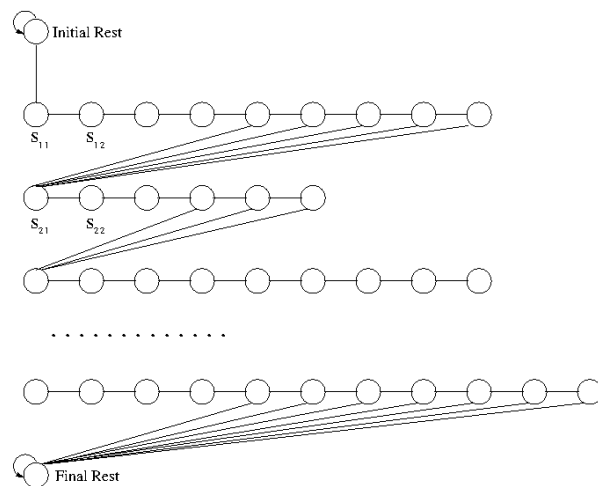


Figure 1. A possible left-to-right graph topology for score alignment.

#### 2.1.1 Transition Probability

Suppose we let  $Q(x, x')$  be the transition probability matrix for  $X$ ,  $Q(x, x') = P(X_{n+1} = x' | X_n = x)$ , where we assume time homogeneity — these probabilities don't depend on  $n$ . Suppose  $L_k$  is the random length of the  $k$ th note and that we have some desired distribution for this length,  $P(L_k = l)$ , which is indicated from the music score. Since

visiting state  $s_{k,m}$  means that the realization of the  $k$ th note is at least  $m$  frames long, we have

$$\begin{aligned} Q(s_{k,m}, s_{k,m+1}) &= P(L_k \geq m+1 | L_k \geq m) \\ &= \frac{\sum_{l=m+1}^M P(L_k = l)}{\sum_{l=m}^M P(L_k = l)}, \end{aligned}$$

and  $Q(s_{k,m}, s_{k+1,1}) = 1 - Q(s_{k,m}, s_{k,m+1})$ . As Figure 1 shows, we append a start state and an end state, both with self-loops, to the beginning and end of the graph as the simple state-space model for the Markov chain,  $X$ . For longer notes, we also allow their states to have self-loops. This model can also be regarded as a hidden semi-Markov model [15] if we view all the “micro states” of a note as one super state.

### 2.1.2 Data Model

We only briefly describe the data model here because it is not the most important part of our proposed idea — one can easily replace our data model with a new one while using our framework. For each frame,  $n$ , we observe a short burst of audio data,  $y_n$ . We model the data likelihood in terms of the normalized magnitude spectrum of  $y_n$ ,

$$e_n(\omega) = \frac{|z_n(\omega)|}{\sum_{\omega'} |z_n(\omega')|} \quad (1)$$

for  $\omega = 1, \dots, \Omega$  where  $z_n$  is the windowed finite Fourier transform of  $y_n$ , and  $\Omega$  is the number of *bins* in the frequency domain. We then model the data likelihood as

$$P(y_n | X_n = x) = \prod_{\omega=1}^{\Omega} q_x(\omega)^{e_n(\omega)} \quad (2)$$

where  $q_x$  is the probability distribution over frequency we associate with state  $x$  (the template of state  $x$ ). Refer to Raphael [16] for detailed description of this data model.

### 2.1.3 Inference

With our HMM in place, it is possible to compute a number of quantities relevant to inference about the audio performance [17]. For instance, one can compute the forward model, giving the evolving state of knowledge on score position,  $P(X_n = x_n | y_{1:n})$  where  $y_{1:n} = y_1, \dots, y_n$ ;  $P(X_n = x_n | y_{1:N})$ , the state distributions given the entire data  $y_{1:N}$ ; or the most likely sequence of states,  $\hat{x} = \arg \max_{x_{1:N}} P(X = x_{1:N} | y_{1:N})$ . All of these computations use dynamic programming or dynamic-programming-like algorithms.

## 2.2 Score-alignment-with-skips and Motivations

Models like the one depicted in Figure 1 assume the player will play the score as written, from the beginning of the excerpt to the end — the usual assumption of score alignment, which is appropriate for many applications, but not reasonable for the “free practice” case at hand. In score-alignment-with-skips, we expect that the player will play *sections* of the score, perhaps repeating them numerous times, before moving on to other sections. When a particular section is practiced, we assume the player will play

the score notes in order (just as in traditional score alignment), according to the notated rhythm. Therefore, we do not wish to completely abandon the basic model of Figure 1.

In our approach, we want to allow occasional skips in which our Markov model,  $X$ , jumps from one score position to another. In practice, the overwhelming majority of these skips are “do overs” — cases where the player repeats a group of notes that are *most recently* played because he or she is unsatisfied with the sound, perhaps repeating numerous times. Therefore, small backward skips are the most likely possibilities. However, we cannot constrain the model to *only* allow such local skips, because occasionally the player will shift to a completely new section of the score, or restart from the beginning. If our model is to be genuinely useful, it must allow for such non-local skips as well. We extend the model of Figure 1 to allow for score skips by adding a “hub” state that communicates in both directions with each of the note models, as proposed in Nakamura et al. [6]; any note can “jump” to or from this hub state.

As discussed earlier, we believe globally optimal “full-fledged” dynamic programming approaches, either for on-line or offline versions of this problem don’t leave sufficient headroom for exploring the space of possible graph topologies or expanding the search space to model *collections* of scores the musician is studying. Thus we focus on *beam search* methods — algorithms that retain a fixed-size list of the currently-best hypotheses at each frame. Typically the beam is several hundred hypotheses in our experiments. Considerations for beam search models are different from those for full-fledged dynamic programming, since a hypothesis must look attractive at *every* stage of the computation in order to avoid being pruned. In addition, we use a “pitch tree” to further help with computation, as will be discussed in what follows.

## 2.3 Pitch Tree and Beam Search

Figure 2 introduces our *global skips* model which allows the player to jump from any score location to any other score location at any time. In the bottom of this figure, the linear sequence of states is a compact description of the original model of Figure 1. In this linear graph each note has been compressed into a single state for simplicity’s sake. Each of these states can either remain in the current state, move forward in the score, or “escape” to the “wait” state at the root of the tree in the top of the figure. The escape probability is chosen to be small enough so that our model is disinclined to recognize one- or two-note (super short) excerpts, but still capable of identifying them. In essence, we use the tree structure to “sort out” the player’s score position in a computationally efficient manner when a jump is made.

The root of the tree is a state with a self loop, modeling the typical pause that occurs as one stops playing and resumes again at a new location. Therefore, the data model for this state is the silence model. The rest of the tree is illustrated in the case of the short “toy” score represented by the pitch sequence  $a, b, c, a, b, a, b$  given in Figure 2. The



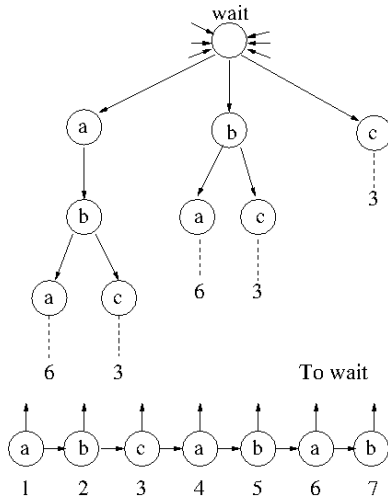


Figure 2. The global skips model in which a “pitch tree” allows the efficient sorting out of the score location after a jump.

possible pitches in the score are  $a$ ,  $b$  and  $c$ , so these define the first level of the tree.  $c$  appears only once in the score so unambiguously defines the position as score note 3. On the other hand,  $b$  has two possible successors in the score ( $bc$  and  $ba$ ), thus two children in the tree. Because each of the children unambiguously identifies a score position (6 or 3), they terminate their branches. The construction continues in this manner until the score position is uniquely identified.

In general, we define a tree construction as follows. Suppose for now we treat a monophonic instrument, although we generalize to polyphonic cases too in our experiments. Let  $M$  be the distinct possible pitches that occur in the score, expressed as the letter names  $\{a, b, c\}$  in Figure 2. We denote the finite-length sequences of such pitches by  $M^*$ . We let  $\tilde{M} \subset M^*$  be the collection of all subsequences of pitches, without regard for rhythm, that appear *multiple* times in the score.  $\tilde{M}$  indexes the non-terminal nodes of our tree:  $\{t_c : c \in \tilde{M}\}$ . We let  $\tilde{M}_0$  denote the pitch subsequences that appear *only once* in the score whose prefixes are in  $\tilde{M}$  — these are the shortest sequences that uniquely determine the score position.  $\tilde{M}_0$  indexes the terminal nodes of our tree:  $\{t_c : c \in \tilde{M}_0\}$ . If  $c \in \tilde{M}$  and  $c' \in \tilde{M} \cup \tilde{M}_0$ , with  $c' = c \circ m$  for  $m \in M$  (sequence  $c'$  is  $c$  concatenated by pitch  $m$ ), then  $t_c \rightarrow t_{c'}$  in the tree graph (non-terminal node  $t_c$  has a successor node,  $t_{c'}$ ). The terminal nodes in the tree are really proxies for the score notes in the linear graph. That is, if  $t_c \rightarrow t_{c'}$  with  $c' \in \tilde{M}_0$ , then  $t_c$  really connects to the score note uniquely identified by the string  $c'$ . This association is made explicit by the dotted lines in Figure 2. For example, in the left branch (out of the three branches), the second level  $b$  really connects to score positions 6 and 3.

The purpose of the tree is efficient computation while maintaining accuracy. After a jump is made, our data model usually argues strongly for a small number of world of possible pitches. Especially in the context of beam search where only the best several hundred hypotheses are kept

Composer	Piece	Meas.	Min.
Mozart	Clarinet Concerto, Mvmt 1	1-154	9
Mozart	Clarinet Concerto, Mvmt 2	1-59	6
Mozart	Clarinet Concerto, Mvmt 3	1-112	13
Brahms	Sonata for Viola in Eb, Mvmt 1	1-97	14
Hoffmeister	Concerto for Viola, Mvmt 1	1-150	11
Bartok	Concerto for Viola	1-200	15
Mozart	Piano Sonata K. 330, Mvmt 1	1-88	14
Beethoven	Piano Sonata op. 110, Mvmt 1	1-70	15
Debussy	<i>La Fille aux Cheveux de Lin</i>	1-39	17

Table 1. List of repertoire used in the experiments.

at each frame, it seems wasteful to maintain an individual hypothesis for each separate place a pitch occurs in the score, so our tree efficiently maintains only a single hypothesis for all such score positions. When the next note is played, we drop down a level in the tree, reaching a state associated with all of the score positions where the pair of pitches occur in order. We continue this process until the score position is unambiguously determined, at which point it “joins” our original score model (as in Figure 1).

In reality, we would not move down the tree deterministically, but, rather, would consider a range of possible tree positions supported by the data model, as is always the case when finding the most likely state sequence of an HMM using dynamic programming and a beam search.

Although beam search does not guarantee a global optimal result, in practice the correct hypotheses usually survives pruning because the data model is strong. The pitch tree also helps with avoiding unwanted prunings, since when a jump is performed we represent the possible score positions compactly, refining our representation as more information becomes available. Even if all correct hypotheses are pruned out at some unfortunate frame, the search can “recover” at any time by jumping to the wait state and then to the correct score position. This behavior is verified in the following experiments.

### 3. EXPERIMENTS

#### 3.1 Data

We collected practice audio from a number of students and faculty mostly in the Jacobs School of Music at Indiana University. These consisted of three undergraduate clarinet majors, three undergraduate viola majors, one faculty pianist, and one student pianist who was not a music major. The data together account for a little less than two hours of practice audio. Part of our goal in collecting these data was to understand the range of variation encountered in real-life practice sessions. In particular, we want to know if the score-alignment-with-skips model of a practice session is tenable — can musicians naturally confine their practice to the score as our model assumes? We also want to know the accuracy of our approach in tracking the players’ score trajectories.

We instructed our subjects to practice an agreed-upon piece of music and only this piece for the duration of their recorded practice session, generally 10 to 20 minutes in a single “take.” Table 1 lists the pieces we tested, along with

the associated measure ranges and practice session lengths. Aside from requesting that the pianists practice with both hands together throughout, we tried to give subjects a minimum of direction and did not supervise their practice or try to otherwise constrain their practice beyond the initial instructions. We observed a number of departures from the basic playing-with-skips assumption, including, for example, deliberately distorted rhythms, testing of reeds, playing significantly slower than the generally-accepted tempo, one-hand piano practice, and brief forays into interval tuning practice only loosely related to the score.

Pieces with verbatim repetition of passages pose problems for both recognition and evaluation: if a player jumps to a passage that appears multiple times in a piece, how can we say for sure which version is being played? However, this distinction doesn't seem especially important from the standpoint of evaluation. For instance, consider the 3rd movement (*Rondo*) of the Mozart Clarinet Concerto, where the refrain repeats six times with almost no variation. It doesn't seem reasonable to penalize our algorithm for failing to ascertain which repetition is being practiced, nor does it seem feasible to make this determination while creating ground truth. For simplicity sake, we chose excerpts from the pieces where there was no direct phrase-level repetition of musical material (as in Table 1).

To create ground truth for an audio example, we first split it by hand into contiguous sections, each section containing music played without skips. We then performed score alignment on each of these sections individually. The results include the section information (the starting note and the starting frame of a section), and onsets of all played notes. The results were then meticulously corrected by hand to be as precise as we could get them.

### 3.2 Evaluation Method and Results

We propose a simple way to evaluate the score-alignment-with-skips problem that is easy to implement and useful for comparing with other approaches. In both ground truth and recognized results, all frames between two consecutive notes are associated with the former note. In other words, any given frame is associated with the note whose onset is the most recent. For every frame, we calculate the “musical distance” between the *recognized* note of this frame and the *ground truth* note of this frame. For example, assuming the time signature is 4/4 and the score has a quarter note at every beat, the musical distance between the second quarter note of measure one and the first quarter note of measure two is 3/4. Such distances tell us the quality of our recognition — how far away the recognized note is from the actual played note. We evaluate our algorithm by counting the number (proportion) of frames that have different levels of musical distances (errors). Our goal is to have more frames in the “0” category (accurate), and fewer frames in the “> 1” category (larger error). Figure 3-5 show this kind of evaluation result for nine pieces in our experiments.

Figure 3 gives these frame-by-frame position errors for the three sessions from the Mozart Clarinet Concerto. This histogram, as well as those of the other two instruments

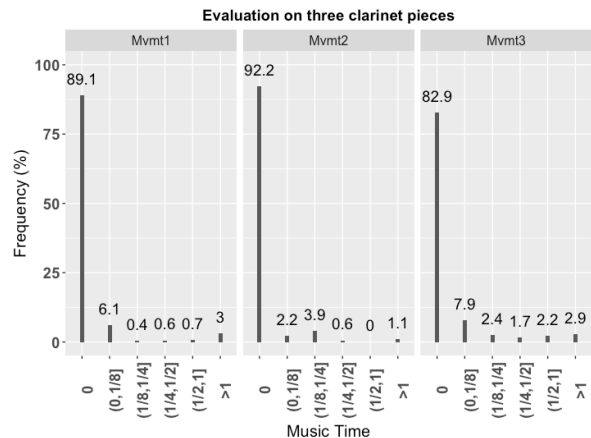


Figure 3. Histograms of frame-by-frame errors for the three practice sessions taken from the three movements of the Mozart Clarinet Concerto, as described in Table 1.

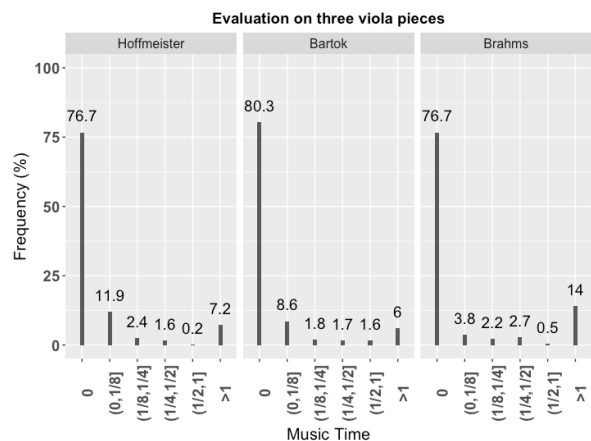


Figure 4. Histograms of frame-by-frame errors for the viola data as described in Table 1.

presented later, bins the errors into several categories generated with split points given as 0, one eighth note, one quarter note, one half note, and one whole note. We use the same binning procedure regardless of the tempo of a piece or its time signature, so, for instance, a whole note error in 6/8 time corresponds to  $1+1/3$  measures. The most important categories are the two extreme ones: “0”, where the score position has been identified as accurately as possible, and “> 1 (whole note)”, where the recognizer is essentially lost. The clarinet is perhaps the easiest instrument to recognize due to its comparative pitch stability. Their results were the best that we measured, with the recognizer lost (error > 1) no more than 3% of the time in all cases. It was interesting to note that the deliberately distorted rhythms observed occasionally in the first movement did not create any problems for recognition.

Figure 4 shows analogous results for the viola data, in which we observed the recognizer being lost from 6% to

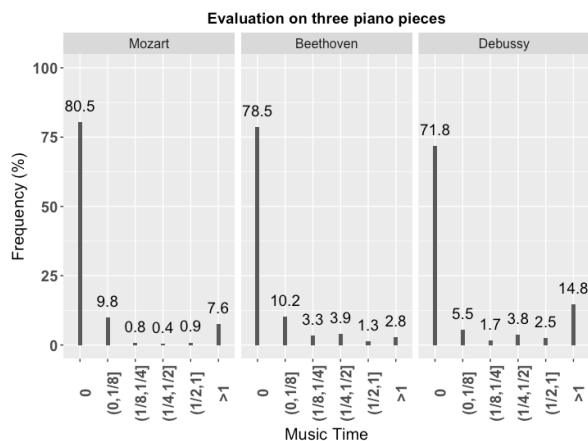


Figure 5. Histograms of frame-by-frame errors for the piano data as described in Table 1.

14% of the time. These results are not quite as good as with the clarinet data, for which we conjecture several reasons. First of all, the viola is simply harder to recognize, since (viola jokes aside) the instrument does not commit itself as clearly to pitch as the clarinet does. In addition, the viola plays double stops (quite a few in some sections of the Bartok and Hoffmeister), while our pitch (data) models tend not to discriminate as well between such chords. Finally, the practice session for the Brahms included many one- or two-note excerpts, and in a couple of cases seemed to be only *inspired* by the score, rather than directly from the score. However, as can be seen from the numerical results, the problems caused by all of these factors were only local.

Figure 5 describes the results for the three piano examples in Table 1. The piano constitutes a significantly harder challenge than the two mostly monophonic instruments. We take a *homophonic* view of the piano, regarding the score as a sequence of chords without regard for voice. That is, whenever the score indicates that a note enters or exits, we create a new chord at the appropriate musical position. This allows the piano to be recognized in the same fashion as used for the other instruments. It should be noted that the “pitch tree” approach of Figure 2 may be less effective as there is far less repetition of chord sequences than of individual pitch sequences.

Generally speaking, the piano is much more challenging than monophonic instruments, since, as mentioned before, the data model discriminates less well between chords than single notes. In addition, the nature of the instrument produces much more overlap between notes, either through pedaling, which is not reflected in our scores, or through the fact that the addition of new notes has no damping effect on preceding notes. In contrast, the essential physics of string, wind, and brass instruments cause the new note to damp the previous note (except in the case of different strings). However, these challenges seem to manifest mostly themselves in terms of small onset inaccuracies, rather than causing the recognition to become lost any more

than with the viola. Again, the “lost” percentages were in the 2% to 14% range. The highest error rate was from the slow Debussy piece, where far less is known about the timing of a performance than with fast music.

A less numerical, but perhaps more illuminating example can be seen at <http://music.informatics.indiana.edu/papers/smc19-skips/>, where the video highlights the player’s current position in a musical score as the practice audio plays. Similar videos for all of the practice sessions can be found at the same web site. In addition to providing an easily digestible demonstration of the heart of this research, the videos also foreshadow the kinds of tools we envision developing for musicians to help review practice effectively.

The results presented here are “exploratory” (on a small dataset), so we obviously cannot claim broad coverage of the world of possible practice habits — such a data collection would be a large undertaking in and of itself. Still, we encountered a good deal of variation within the sampled population. We believe the results show that our essential practice assumptions are reasonable, in the sense that our subjects were, for the most part, able to follow our model of score-constrained practice without much difficulty, while cases that departed from our model created only local problems or no problems at all. We believe the accuracy of our score alignment is also promising. In short, the algorithm occasionally gets lost but always finds its way back to the correct score position. Furthermore, we believe the accuracy of note onset estimates on the individual identified sections is certainly good enough for many kinds of pedagogical feedback.

#### 4. FUTURE WORK

The basic recognition ideas developed here can be embedded into thought-provoking and illuminating tools to help instrumentalists review their practice, along the lines of the demonstrated video. To achieve this goal we must both improve the basic recognition on which these “practice browsers” will rely, as well as identify forms of feedback of interest to students, and ways of expressing that feedback visually.

The current approach assumes one cannot tell the difference between identical passages in a piece of music, though this is only partly true. Typically we can make rather strong assumptions about the way in which the musician will visit the piece of music. In particular, nearly all jumps are local ones, and the overwhelming majority of jumps move backwards. Nakamura et al. [18] also analyzed the skipping property on three piano pieces. These assumptions do not fit naturally with our sorting trees of Figure 2 since the non-terminal nodes of the tree are associated with multiple score positions. Therefore, we can’t assess their distance from the jump origin. In contrast, a simple model that allows only local backward jumps performs surprisingly well for the overwhelming majority of cases. However, when the local backward assumption is violated this model becomes completely lost, thus is too fragile. We anticipate that it is possible to find a modeling framework that can express the likelihood of various jumps, while also retaining the computational efficiency of

our sorting tree. This is a project for future study.

Tuning is certainly an obvious candidate for visualization, perhaps by coloring score notes according to the tuning error. Coverage of a practice session could be similarly represented, using color to denote the number of times a particular passage has been repeated. Giving feedback on rhythm is more challenging, partly because there will always be some degree of error in the identification of note onsets, but also because good rhythm depends both on timing and stress (or lack of stress). We anticipate considerably challenge here, though clearly there is much fertile ground to explore.

## 5. REFERENCES

- [1] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *ICMC*, vol. 84, 1984, pp. 193–198.
- [2] B. Vercoe, "The synthetic performer in the context of live performance," in *Proceedings of International Computer Music Conference*, 1984, pp. 199–200.
- [3] P. Cuvillier, "On temporal coherency of probabilistic models for audio-to-score alignment," Ph.D. dissertation, Université Pierre et Marie Curie-Paris VI, 2016.
- [4] A. Cont, D. Schwarz, and N. Schnell, "Training ircam's score follower [audio to musical score alignment system]," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*, vol. 3. IEEE, 2005, pp. iii–253.
- [5] D. Schwarz, A. Cont, and N. Schnell, "From boulez to ballads: Training ircam's score follower," in *International Computer Music Conference (ICMC)*, 2005, pp. 1–1.
- [6] T. Nakamura, E. Nakamura, and S. Sagayama, "Real-time audio-to-score alignment of music performances containing errors and arbitrary repeats and skips," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 2, pp. 329–339, 2016.
- [7] M. Puckette, "Score following using the sung voice," in *ICMC*, 1995.
- [8] L. Grubb and R. B. Dannenberg, "A stochastic method of tracking a vocal performer," in *ICMC*, 1997.
- [9] C. Raphael, "Automatic segmentation of acoustic musical signals using hidden markov models," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 4, pp. 360–370, 1999.
- [10] R. J. Turetsky and D. P. Ellis, "Ground-truth transcriptions of real music from force-aligned midi syntheses," 2003.
- [11] N. Orio and F. Déchelle, "Score following using spectral analysis and hidden markov models," in *ICMC: International Computer Music Conference*, 2000, pp. 1–1.
- [12] P. Cano, A. Loscos, and J. Bonada, "Score-performance matching using hmms," in *ICMC*, 1999.
- [13] M. Müller and D. Appelt, "Path-constrained partial music synchronization," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 65–68.
- [14] C. Raphael, "Music plus one and machine learning," in *ICML*, 2010, pp. 21–28.
- [15] K. P. Murphy, "Hidden semi-markov models (hsmms)," *unpublished notes*, vol. 2, 2002.
- [16] C. Raphael, "A hybrid graphical model for aligning polyphonic audio with musical scores," in *ISMIR*. Citeseer, 2004, pp. 387–394.
- [17] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
- [18] E. Nakamura, T. Nakamura, Y. Saito, N. Ono, and S. Sagayama, "Outer-product hidden markov model and polyphonic midi score following," *Journal of New Music Research*, vol. 43, no. 2, pp. 183–201, 2014.



# PIANO SCORE-FOLLOWING BY TRACKING NOTE EVOLUTION

**Yucong Jiang**  
Indiana University Bloomington  
yujiang@iu.edu

**Christopher Raphael**  
Indiana University Bloomington  
craphael@indiana.edu

## ABSTRACT

Score following matches musical performance audio with its symbolic score in an on-line fashion. Its applications are meaningful in music practice, performance, education, and composition. This paper focuses on following *piano* music — one of the most challenging cases. Motivated by the time-changing features of a piano note during its lifetime, we propose a new method that models the evolution of a note in spectral space, aiming to provide an adaptive, hence better, data model. This new method is based on a switching Kalman filter in which a hidden layer of continuous variables tracks the energy of the various note harmonics. The result of this method could potentially benefit applications in de-soloing, sound synthesis and virtual scores. This paper also proposes a straightforward evaluation method. We conducted a preliminary experiment on a small dataset of 13 minutes of music, consisting of 15 excerpts of real piano recordings from eight pieces. The results show the promise of this new method.

## 1. INTRODUCTION

Score following matches musical performance audio with its symbolic score, as illustrated in Figure 1. This paper focuses on following *piano* music, which is one of the most challenging score following cases, due to the high degree of polyphony in the piano. We restrict our attention to the *on-line* version of the problem which allows no “look ahead” in the audio, as is appropriate for real-time applications.

Score following is the foundation of many useful applications. It forms the heart of any musical score page turner [1], as well as a crucial layer of automatic accompaniment systems [2]. For composers, it enables virtual scores, scores that consist of electronic programs that react to a live performance [3]. Using the recognized score information during a performance, a score follower can give feedback concerning the performance at the signal level, which can be further developed into a music-education tool, e.g., a computer tutor [4]. It can also serve real-time audio enhancement applications, processing the input audio while outputting the enhanced audio in real-time, e.g., auto-tune in a live performance.

Copyright: © 2019 Yucong Jiang et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

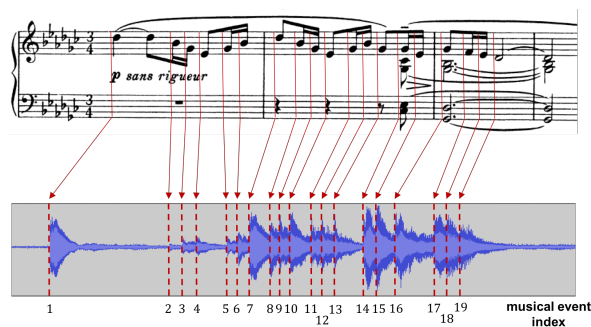


Figure 1: An ideal score-following result of an example excerpt. The dotted red lines are note/chord onsets.

Most methods of score following share the same general idea: possible (hypothesized) performances are viewed as paths through a state graph which is derived from the musical score. For any moment in the audio, we infer the current state of the performance given the available audio data (up to this moment) — following the score according to the played music. However, existing methods differ in how their models *score* these paths.

Many methods are based on the hidden Markov model (HMM) or its variations [5–10], including one of the state-of-the-art systems, Music Plus One [11], which is the *baseline* system in this paper. Another leading system, Antescofo [12], uses a hidden *semi*-Markov model. Some efforts also model the *tempo* in music [13–17]. Refer to Cuvillier [18] for a thorough literature review on this topic. The off-line version of this problem also shares some common techniques with score following [13] [19].

The two state-of-the-art systems mentioned above have been successfully used to follow soloists in live concerts (mostly on monophonic instruments), but are much less robust on *piano* music. Piano music is usually highly polyphonic, with many notes sounding at the same time, making it significantly more difficult to develop a discriminating data model. In addition, pedaling often prolongs notes beyond their nominal offsets in the score, causing mismatches between the audio and the score. For those reasons (among others), piano score following remains an open and unsolved challenge.

The purpose of this paper is to introduce a new approach to *data modeling* in score-following problems, especially for piano music. Existing methods generally assume that a note has a *fixed* data model that is applied to all (or most) frames associated with that note, with the possible exception of the opening frame(s) where the “attack” happens.

However, this assumption is flawed, especially for piano audio. Each piano note decays over time, with significantly different decay rates for different frequencies. This results in a *changing* frequency spectrum over the life of the note. Our current effort models this note-level harmonic evolution. Based on Music Plus One’s HMM framework, we use a *switching Kalman filter* [20] to track the individual amplitudes of the harmonics of each note. This model can adapt to the time-changing features of a note, providing, we hope, a more discriminating data model.

There are applications that could potentially benefit from the *tracked amplitudes* as part of our score following results. One example would be de-soloing, where the precise model of the data could be used to “subtract” or remove it from a recording with other instruments. In modeling the piano sound for synthesis, such amplitude information could also be useful. In addition, the tracked amplitudes can provide valuable information for virtual score related applications — for example, triggering a program when a harmonic decreases below a certain threshold.

## 2. REPRESENTING SCORE

We simplify a musical score as a sequence of chords (Figure 2), essentially adopting a homophonic view of the music. This way, polyphonic music can be represented *linearly* as a sequence of event pairs: {musical time, note(s)}. We simply refer to these events as “chords” in this paper.



Figure 2: “Homophonic” view of polyphonic music (modified from [21]). The left bar is the original score with two voices. The right bar represents this score as a sequence of chords.

## 3. METHOD

In this section, we first introduce the HMM framework that represents the baseline method. This HMM framework is also the foundation of the proposed new method. We then explain the motivations of our new method and its assumptions, before describing how a Kalman filter tracks a single partial of a chord. Lastly, we explain how the Kalman filter fits into the HMM framework, resulting in a *switching Kalman filter* model that tracks the changing features during the evolution of a note.

### 3.1 HMM Framework

A score-following HMM models the performance as a path through a state graph. The state graph is constructed directly from the score by specifying one or several states for each note (or chord), while forcing left-to-right movement through these states. We model time as a sequence of audio “frames”, each frame about 64 ms. long. We denote the state process, modeled as a Markov chain, by  $X_1, X_2, \dots, X_T$  where  $T$  is the total number of frames, as in Figure 3. If  $X_t = x_t$ , for some graph state  $x_t$ , we

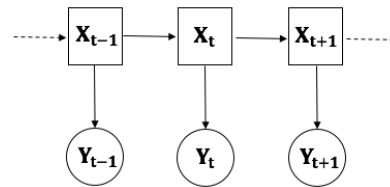


Figure 3: HMM in the baseline model. Squares are discrete variables; circles are continuous variables.

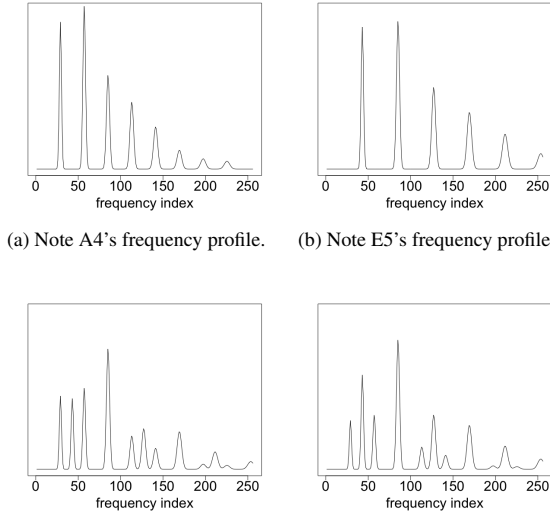
denote its corresponding chord index by  $C(x_t)$ . The state graph, along with the transition probabilities in it, model the *timing* information given by the score. They are also called the *prior model* because they represent our knowledge about the states *before* observing any data. Since the prior (or timing) model is not the focus of this paper, we refer to Raphael [11] for further details about how the state graph and transition probabilities are designed.

The other part of the HMM framework is the *data model* (our focus) — how we score each hypothesis state according to the observed data. The observed data vector for each frame is the magnitude Fourier spectrum of the corresponding frame of data, normalized to sum to 1. Let vector  $\mathbf{y}_t$  be this observed feature at frame  $t$ ,  $\mathbf{y}_t = y_t^1, \dots, y_t^K$ , and let  $\mathbf{q}_i = q_i^1, \dots, q_i^K$  be the template of chord  $i$  with the same dimension,  $K$ . The template is also normalized to sum to 1, thus representing it as a probability distribution. If we view the feature vector  $\mathbf{y}_t$  as the histogram of a random sample from  $\mathbf{q}_i$ , the likelihood of observing this feature given the state is a multinomial distribution (the eliminated constant coefficient is irrelevant for comparing different hypotheses because the data is fixed):

$$P(Y_t = \mathbf{y}_t | X_t = x_t) = \prod_{k=1}^K q_i^k y_t^k,$$

where  $i = C(x_t)$ .

The template is a mixture of all notes involved in a chord, and each note is composed of a Gaussian mixture, one component for each harmonic of the note. For example, Figure 4 shows the template of a single note A4 or E5, along with two possible templates of the chord “A4 and E5”. In the baseline model, the template for each chord,  $\mathbf{q}_i$ , is carefully calibrated, but *fixed* — it cannot adapt to the given data once it is (pre)defined. In other words, the baseline model assumes that the (normalized) spectrum of a chord can be expected before observing the data, and that it does not change over the lifetime of the chord (from the onset frame until the onset of the next chord). However, in fact, we do not know the relative ratio of the notes in a chord beforehand, thus cannot accurately anticipate the template of this chord (e.g., c and d in Figure 4), and the chord’s spectrum *does* evolve over time. We propose a new method that uses *flexible* templates, allowing them to adapt to the changing energy distribution among harmonics during a chord’s lifetime.



(c) The frequency profile of A4 mixed with E5 by the ratio of 1:1. (d) The frequency profile of A4 mixed with E5 by the ratio of 1:2.

Figure 4: a and b are the frequency profiles of two different notes. c and d are the frequency profiles of a two-note chord, played with two different relative loudness ratios.

### 3.2 Motivations and Assumptions

The piano is a percussion instrument; the sound of each note decays over time, in sharp contrast to instruments like the violin where the entire evolution of each note remains under the player's control. The rate of decay differs among different partials, with higher frequency partials usually decaying faster than the lower ones, thus leading to a *changing* spectrum in the same chord. The baseline method, however, cannot capture this phenomenon, because all frames within a chord share the same fixed spectrum template. This causes a mismatch between the data templates and the real observed data. In contrast, our proposed method updates the spectrum template at every frame after observing the newly received data.

Given a specific chord, we know *where* its partials lie in the frequency domain, though we are not sure about their relative intensities. We model every partial in the frequency domain with the shape of a truncated (and discretized) Gaussian density function, as in the left column of Figure 5. We divide the template into multiple frequency regions according to the location of the partials, each region completely separate from the others (by the dotted line in Figure 5). To make the computation plausible, the data model assumes that the amplitudes governing the different harmonics are conditionally independent, given the state. That is to say, for a single chord hypothesis, there might be a collection of neighboring frames associated with this hypothesis, and the harmonics are assumed to evolve independently from each other in these frames. We have also considered the inharmonicity of a piano when constructing partials.

Some of these partials might overlap in frequency; it is common for harmonics from different notes to “collide”

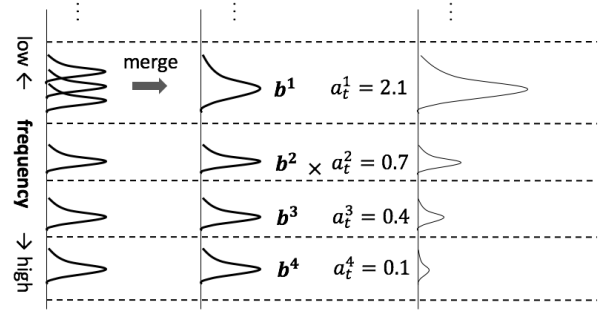


Figure 5: Demonstration of partials. Left: the original structure of six partials; Middle: the partial structure of four independent partials after merging; Right: a data template which is a superposition of four partials with different amplitudes. Each region divided by the dotted lines corresponds to one independent Kalman filter.

at the same frequency, creating an *identifiability* problem in distinguishing their amplitudes. In practice, we address this by merging them, and treat them collectively as a single partial that also has the shape of a truncated Gaussian, and with the same frequency boundary as the group (middle column of Figure 5).

### 3.3 An Independent Kalman Filter

We use a Kalman filter, independently for every region, to track the amplitude of the partial. The independence of the Kalman filters is justified by the assumption that the partials have non-overlapping support. This section describes how a single Kalman filter tracks the amplitude of one partial (including merged partials) over the lifetime of a chord. Let's look at the  $p$ th partial of a chord. The shape of this partial is denoted by  $\mathbf{b}^p$ , which is truncated within a limited range of frequencies, as in the left column of Figure 6.  $\mathbf{b}^p$  is a constant vector and sums up to 1. Denote this partial's amplitude at frame  $t$  as  $a_t^p$ , and assume  $a_t^p \sim \mathcal{N}(m_t^p, v_t^p)$  — a normal distribution with mean  $m_t^p$  and variance  $v_t^p$ . The amplitude decays with time, decaying exponentially at rate  $\lambda (< 1)$ , perhaps depending on the frequency. The Kalman filter models this decay as

$$a_t^p = \lambda a_{t-1}^p + \epsilon_t^p,$$

where  $\epsilon_t^p \sim \mathcal{N}(0, \sigma^2)$ . Figure 6 shows the decay of a partial over three frames. Note that we are not tracking the amplitude of every frequency, but the amplitude of the truncated Gaussian,  $\mathbf{b}^p$ . For example, at given time  $t$ , the most likely spectrum of this partial is  $m_t^p \cdot \mathbf{b}^p$ , which is still a truncated Gaussian.

At this frame, the predicted observation is modeled as

$$\mathbf{y}_t^p = a_t^p \cdot \mathbf{b}^p + \delta_t^p,$$

where the components of  $\delta_t^p$  are independent 0-mean Gaussian noise:  $\delta_t^p \sim \mathcal{N}(0, \rho^2 I)$ , and  $\mathbf{y}_t^p$  is the observed data within the frequency range of partial  $p$ . Under these assumptions, the Kalman filter provides a straightforward update equation computing the conditional distribution on

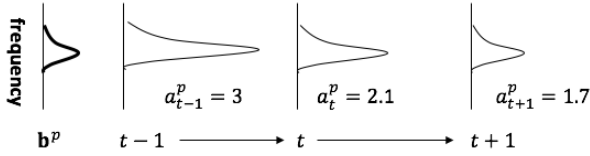


Figure 6: The shape of a partial (left), and its amplitude's decay between neighboring frames.

$p(a_t^p | \mathbf{y}_{1:t}, \dots, \mathbf{y}_{t:t})$ . Note that the observed data at frame  $t$ ,  $\mathbf{y}_t$ , is a superposition of all members in  $\{\mathbf{y}_t^p\}$ .

### 3.4 Switching Kalman Filter

This section describes how the Kalman filters for tracking individual partials extend the filtering framework of our HMM. In Figure 7, the newly added middle layer of variables (cf. Figure 3), notated as  $A$ , represents the amplitude information of *all* partials in state  $X$ 's corresponding chord. A chord has multiple partials,  $\{a_t^p\}$ ,  $p = 1, \dots, P$ , each of which is tracked by an independent Kalman filter as described above. For example, in Figure 5, each of the four partials is tracked by an independent Kalman filter. At each frame, we update a chord hypothesis' template by the tracked amplitudes of its partials, resulting in models that are better adapted to the audio data.

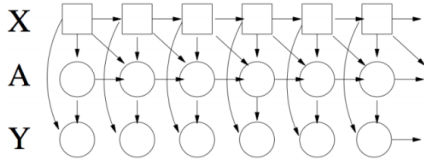


Figure 7: Directed acyclic graph showing the conditional independence structure of the model variables.  $X$  and  $A$  are hidden variables;  $Y$  is observable variables. Squares are discrete variables; circles are continuous variables.

At every frame, there will be multiple hypotheses concerning its position in the score,  $X_t$ . Each hypothesis has an associated Gaussian distribution for each of the chord's partials. Therefore, writing  $\mathbf{y}_{1:t}$  for  $\mathbf{y}_1, \dots, \mathbf{y}_t$ , our representation of the *filtered* distribution (the distribution on the hidden variables at time  $t$  after observing the data up to time  $t$ ) is

$$p(x_t, \mathbf{a}_t | \mathbf{y}_{1:t}) = p(x_t | \mathbf{y}_{1:t}) \prod_{p=1}^P p(a_t^p | x_t, \mathbf{y}_{1:t}). \quad (1)$$

In contrast with the HMM filtered distribution, our switching Kalman filtered distribution is given by a discrete state probability,  $p(x_t | \mathbf{y}_{1:t})$ , and a product of Gaussian densities on the  $\{a_t^p\}$ , for each hypothesis on the current discrete state,  $x_t$ .

From frame  $t$  to  $t+1$ , the amplitudes evolve in two different styles depending on the values of the new state — the new state either remains in the same chord,  $C(x_{t+1}) = C(x_t)$ , or must move to the subsequent chord,  $C(x_{t+1}) = C(x_t) + 1$ . In the former case, the structure of the partials

doesn't change, and all partials simply follow the evolution process described in Section 3.3. In the latter case, a new chord is starting, with a different harmonic structure from the previous chord. We assume that any *new* partials are initialized from a default distribution,  $a_{t+1}^p \sim \mathcal{N}(m_0, v_0)$ , where  $m_0$  and  $v_0$  are the initial mean and variance of a partial's amplitude, while any *continuing* partials from the previous chord simply evolve according to the Kalman filter model in Section 3.3.

#### 3.4.1 Inference

This section explains how the filtered distribution of the two hidden variables,  $x_t$  and  $\mathbf{a}_t$  (as in Equation (1)), evolves as it goes from  $t$  to  $t+1$ . It includes two steps: after conditioning on the new data observation,  $\mathbf{y}_{t+1}$ , it marginalizes out the partial amplitudes,  $\mathbf{a}_t$ ; then, it marginalizes out the state,  $x_t$ . These two steps will be represented in Equation (2) and Equation (3) respectively.

Let's look at the amplitudes first. As mentioned in Section 3.4, the amplitudes evolve in two different styles. In the case of a continuing chord, we can compute the probability

$$p(x_{t+1}, x_t, \mathbf{a}_{t+1} | \mathbf{y}_{1:t+1}) = \prod_p p(x_{t+1}, x_t, a_{t+1}^p | \mathbf{y}_{1:t+1}) \quad (2)$$

according to the usual update formula of the Kalman filter, applied independently to each partial,  $a_{t+1}^p$ . In doing so, the distribution for each partial  $a_{t+1}^p$  is conditioned on the relevant (and non-overlapping) portion of  $\mathbf{y}_{t+1}$ . In the other case, if it is a new chord, the amplitudes of new partials adopt the default distribution  $\mathcal{N}(m_0, v_0)$ , and the continuing partials follow the same process as in the case of a continuing chord. Therefore, in the latter case, too, we can compute the value of Equation (2) in a straightforward manner.

The above discussion shows how to marginalize out the continuous variables  $\{a_t^p\}$  through the standard Kalman filter formulation. The difficulties of implementing a switching Kalman filter arise when we further marginalize out the discrete variable,  $x_t$ , by

$$p(x_{t+1}, \mathbf{a}_{t+1} | \mathbf{y}_{1:t+1}) = \sum_{x_t} p(x_{t+1}, x_t, \mathbf{a}_{t+1} | \mathbf{y}_{1:t+1}). \quad (3)$$

The difficulty is that different predecessors,  $x_t$ , are associated with different estimates of  $\mathbf{a}_{t+1}$ . When summing out all possible predecessors, the estimate of each partial's amplitude becomes a Gaussian mixture model. The number of components grows exponentially with  $t$ , making the problem intractable without approximation. We use an approach familiar in the switching Kalman filter literature, approximating the mixture of multiple Gaussians by a single Gaussian with the same mean and variance as in the mixture [20]. Say the  $i$ th element in the mixture is a Gaussian  $\mathcal{N}(m_i, v_i)$ , and has probability  $p_i$  (mixture weight). The approximated Gaussian, then, has mean and variance:

$$m = \frac{1}{\sum_i p_i} \cdot \sum_i p_i \cdot m_i$$



$$v = \frac{1}{\sum_i p_i} \cdot \sum_i \{ p_i \cdot v_i + p_i \cdot (m_i - m)^2 \}$$

Figure 8 demonstrates this process.

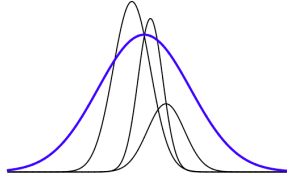


Figure 8: Approximation of a mixture of three Gaussians by a single Gaussian (with thicker blue line).

#### 4. PRELIMINARY EXPERIMENT

We conducted a preliminary experiment on a small dataset. The purpose of this experiment was to benchmark the initial development of this new method (with a state-of-the-art system), and to inspire further discussions about this direction. We also propose a new evaluation method.

##### 4.1 Data and Settings

The dataset consists of 15 excerpts of real piano performance recordings of eight pieces, detailed in Table 1. Each excerpt lasts about 45 seconds on average, making the dataset 13 minutes in total. The audio is sampled at rate 8 kHz. The frame length is 512 samples (64 milliseconds), and the hop size is 256 samples. There are five important parameters in our proposed method (cf. Sections 3.3 and 3.4.1): the decay factor, the variance of the process noise, the variance of the observation noise, and the initial mean and variance of partials' amplitudes when a new partial comes into existence. They were manually set in this experiment. The audio data are available at <http://music.informatics.indiana.edu/papers/smc19-evolution/>.

Index	Composer	Piece	Measures
1	Mozart	Piano Concerto No.17 in G major, mvmt1	74 - 94
2	Mozart	Piano Concerto No.17 in G major, mvmt1	139 - 171
3	Mozart	Piano Concerto No.17 in G major, mvmt1	184 - 207
4	Schumann	Piano Concerto in A minor, mvmt1	1 - 4
5	Schumann	Piano Concerto in A minor, mvmt1	11 - 19
6	Schumann	Piano Concerto in A minor, mvmt1	58 - 67
7	Chopin	Barcarolle, Op.60	1 - 9 (1)
8	Chopin	Barcarolle, Op.60	1 - 9 (2)
9	Chopin	Prelude, Op. 28 No. 4	1 - 12
10	Chopin	Prelude, Op. 28 No. 4	13 - 26
11	Schubert	Six Moments, D. 780 No. 2	1 - 17
12	Schubert	Six Moments, D. 780 No. 2	18 - 36
13	Debussy	Prelude, No. 2 (Voiles)	1 - 21
14	Beethoven	Piano Sonata, No. 8 (Sonata Pathétique)	1 - 10
15	J.S. Bach	Wachet auf, BWV 140	1 - 12

Table 1: Piano excerpts in the experiment. Excerpts No. 7 and No. 8 are different performances of the same music.

#### 4.2 Evaluation Method and Results

Evaluating (on-line) score-following systems is different from evaluating off-line alignment systems, where one could judge the result by simply comparing the detected notes with the ground truth, e.g., counting the mislabeled frames. Here, however, we cannot count mislabeled frames because there is no onset detection that follows directly from the filtered distribution (unlike in Cont et al. [22]).

We propose a new evaluation method that assesses the filtered distributions at each frame in a straightforward manner. The *frame-wise accuracy* is defined as follows. For each frame  $t$ , the filtered approximation contains the distribution  $p(x_t|y_{1:t})$ . Using ground truth, we can compute the probability of the filtered distribution covering the correct chord as

$$Acc_t = \sum_{x_t: C(x_t)=i_t} p(x_t|y_{1:t}),$$

where  $i_t$  is the ground-truth chord index for frame  $t$ . The total measure of the accuracy aggregates this over all frames:

$$Acc = \sum_t Acc_t,$$

which summarizes how well the algorithms perform.

We use one of the state-of-the-art systems, Music Plus One [11], as the baseline, and compare it with our proposed method. Table 2 shows the frame-wise accuracies of the 15 excerpts. The proposed method has 5.7% higher accuracy than the baseline on average. It also beats the baseline more often.

Index	Baseline	Tracking Note Evolution
1	0.78	<b>0.82</b>
2	<b>0.82</b>	0.81
3	0.69	<b>0.72</b>
4	0.71	<b>0.80</b>
5	0.79	<b>0.88</b>
6	0.76	<b>0.80</b>
7	0.71	0.71
8	lost	<b>0.67</b>
9	<b>0.63</b>	0.56
10	0.49	<b>0.50</b>
11	<b>0.86</b>	0.83
12	0.72	0.72
13	0.64	<b>0.71</b>
14	<b>0.72</b>	0.68
15	<b>0.81</b>	0.77
average accuracy	0.675	<b>0.732</b>
<b>win</b>	5 excerpts	8 excerpts
> 5% <b>win</b>	1 excerpt	4 excerpts

Table 2: Comparing frame-wise accuracy between the baseline and the proposed method. The higher accuracy of an excerpt is bolded (italic bolded if higher than 5%). “Lost” means the program failed and the accuracy is smaller than 0.1.

## 5. DISCUSSION

As shown in Table 2, our proposed algorithm achieved 5.7% higher accuracy than the baseline — a state-of-the-art system. This algorithm also successfully followed excerpt No. 8 where the baseline completely failed. We found that this excerpt involves heavier pedaling than others — usually a sign of the more challenging cases. We speculate that the new method can provide a more discriminating data model for one key reason. Both correct *and* incorrect hypotheses can score better by adapting their templates to the data. However, the correct hypotheses have greater potential to adapt *well*, because their templates have the *right* adapting freedom that incorrect hypotheses don't necessarily have. For example, an incorrect hypothesis has to ignore an observed peak because its template lacks the corresponding harmonic of this peak.

Two limitations prevent us from drawing general conclusions about our new method. First, the testing dataset is small. Second, if we drop the excerpt where the baseline failed, the new method beats the baseline by only 1.3%. Therefore, the results are inconclusive. It is possible that the new approach gained only modest improvement on a small sample. However, we think this new model is scientifically interesting and is valuable for inspiring further discovery in this direction.

We should point out that the current version of the model is fairly basic, and still has much potential to be improved. The five important parameters mentioned in Section 4.1 were manually set, but training them could potentially lead to better results. For example, perhaps different frequencies should not share the same decay rate, because higher frequencies usually decay faster than lower ones.

During the experiment, we discovered that pedaling tends to cause *delayed* detection of a chord, because the algorithms can mistake a chord for the previous chord(s) when observing prolonged note(s) mixed with the current chord. To address this issue, we will consider modeling pedaling in future, or including a new feature for detecting a new starting chord — for example, if the *spectrum difference* between neighboring frames is always positive at some frequencies, it indicates that a new chord is starting.

The proposed idea can be generalized to other features besides the spectrum feature. The spirit is to track the time-changing nature of a note during its lifetime, aiming to provide a more accurate and discriminating data model, especially for challenging cases, like the piano. This flexible framework also allows incorporating multiple features, together forming a better data model from different perspectives. Based on the generally positive result and the above discussion, we believe that this direction leads to an unexplored world, a promising path toward tackling some of the most challenging cases in the score-following arena.

## 6. REFERENCES

- [1] A. Arzt, G. Widmer, and S. Dixon, "Automatic page turning for musicians via real-time machine listening." in *ECAI*, 2008, pp. 241–245.
- [2] R. B. Dannenberg and C. Raphael, "Music score alignment and computer accompaniment," *Communications of the ACM*, vol. 49, no. 8, pp. 38–43, 2006.
- [3] A. Cont, "On the creative use of score following and its impact on research," in *SMC 2011: 8th Sound and Music Computing conference*, 2011.
- [4] R. B. Dannenberg, M. Sanchez, A. Joseph, P. Capell, R. Joseph, and R. Saul, "A computer-based multimedia tutor for beginning piano students," *Journal of New Music Research*, vol. 19, no. 2-3, pp. 155–173, 1990.
- [5] C. Raphael, "Automatic segmentation of acoustic musical signals using hidden markov models," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 21, no. 4, pp. 360–370, 1999.
- [6] N. Orio and F. Déchelle, "Score following using spectral analysis and hidden markov models," in *ICMC: International Computer Music Conference*, 2000, pp. 1–1.
- [7] A. Cont, "Realtime audio to score alignment for polyphonic music instruments, using sparse non-negative constraints and hierarchical hmms," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 5. IEEE, 2006, pp. V–V.
- [8] T. Nakamura, E. Nakamura, and S. Sagayama, "Real-time audio-to-score alignment of music performances containing errors and arbitrary repeats and skips," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 2, pp. 329–339, 2016.
- [9] R. B. Dannenberg and N. Hu, "Polyphonic audio matching for score following and intelligent audio editors." in *ICMC*, 2003, pp. 27–34.
- [10] S. Dixon, "Live tracking of musical performances using on-line time warping," in *Proceedings of the 8th International Conference on Digital Audio Effects*. Cite-seer, 2005, pp. 92–97.
- [11] C. Raphael, "Music plus one and machine learning." in *ICML*, 2010, pp. 21–28.
- [12] A. Cont, "Antescofo: Anticipatory synchronization and control of interactive parameters in computer music." in *International Computer Music Conference (ICMC)*, 2008, pp. 33–40.
- [13] C. Raphael, "A hybrid graphical model for aligning polyphonic audio with musical scores." in *ISMIR*. Citeseer, 2004, pp. 387–394.
- [14] N. Montecchio and A. Cont, "A unified approach to real time audio-to-score and audio-to-audio alignment using sequential montecarlo inference techniques," in *ICASSP 2011: Proceedings of International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2011, pp. 193–196.

- [15] T. Otsuka, K. Nakadai, T. Takahashi, T. Ogata, and H. Okuno, "Real-time audio-to-score alignment using particle filter for coplayer music robots," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, no. 1, p. 384651, 2011.
- [16] F. Korzeniowski, F. Krebs, A. Arzt, and G. Widmer, "Tracking rests and tempo changes: Improved score following with particle filters," in *ICMC*, 2013.
- [17] A. Arzt, "Flexible and robust music tracking," Ph.D. dissertation, Ph. D. thesis, Universität Linz, Linz, 2016.
- [18] P. Cuvillier, "On temporal coherency of probabilistic models for audio-to-score alignment," Ph.D. dissertation, Université Pierre et Marie Curie-Paris VI, 2016.
- [19] M. Müller, "Information retrieval for music and motion, chapter 5," 2007.
- [20] K. P. Murphy, "Switching kalman filters," 1998.
- [21] C. Raphael, "Aligning music audio with symbolic scores using a hybrid graphical model," *Machine learning*, vol. 65, no. 2-3, pp. 389–409, 2006.
- [22] A. Cont, D. Schwarz, N. Schnell, and C. Raphael, "Evaluation of real-time audio-to-score alignment," in *International Symposium on Music Information Retrieval (ISMIR)*, 2007.

# Adaptive Score-Following System by Integrating Gaze Information

**Kaede Noto**  
Graduate School of  
Future University Hakodate  
g2118030@fun.ac.jp

**Yoshinari Takegawa**  
Graduate School of  
Future University Hakodate  
yoshi@fun.ac.jp

**Keiji Hirata**  
Graduate School of  
Future University Hakodate  
hirata@fun.ac.jp

## ABSTRACT

In actual piano practice, people of different skill levels exhibit different behaviors, for instance leaping forward or to an upper staff, mis-keying, repeating, and so on. However, many of the conventional score following systems hardly adapt such accidental behaviors depending on individual skill level, because conventional systems usually learn the frequent or general behaviors. We develop a score-following system that can adapt a user's individuality by combining keying information with gaze, because it is well-known that the gaze is a highly reliable means of expressing a performer's thinking. Since it is difficult to collect a large amount of piano performance data reflecting individuality, we employ the framework of the Bayesian inference to adapt individuality. That is, to estimate the user's current position in piano performance, keying and gaze information are integrated into a single Bayesian inference by Gaussian mixture model (GMM). Here, we assume both the keying and gaze information conform to normal distributions. Experimental results show that, taking into account the gaze information, our score-following system can properly cope with repetition and leaping to an upper row of a staff, in particular.

## 1. INTRODUCTION

The goal of our study is to build a score-following system that adapts a user's individuality. Score-following is one of the important topics in MIR and is a fundamental technique used in many applications including automatic accompaniment, estimation of current position from audio [4, 5, 19], and estimation from symbols [2, 3, 16, 17]. In reality, a score-following system may often face problematic situations, in which current position leaps forward or backward freely. Such leaps occur because of repetition and wrong keying, and during practice. Therefore, researchers of score-following attempt to propose systems and/or algorithms for reacting to or tracking a current position which includes occasional leaping forward or backward [10–12, 18]. For a score without repeated and/or iterated phrases, conventional systems and algorithms can estimate current position almost correctly.

Gaze can give us significant information for score-following, when, for instance, starting performance from an arbitrary position [13]. Our system can use gaze information to estimate current position correctly to some extent, even when keying information is unavailable. There is, however, a crucial issue to be considered which is called eye-hand-span (EHS). EHS means the distance between the point on the score at which a player looks to obtain, in advance, information of notes to be played, and the actual current keying position [14]. Usually, during performance, a pianist looks at a point on the score approximately a phrase or a few notes ahead of current keying position. Since the length of EHS depends on, for instance, individuality, the structure of the melody, the degree of proficiency, and tempo, our previous system takes into account the average length of EHS obtained from experimental data and estimates the current position from both gaze and keying information multiplied by fixed weights. Thus, the system unfortunately neither conducts individual EHS estimation nor assigns the optimum weights to both gaze and keying information for each pianist.

This paper proposes a score-following system, which adapts a user's individuality in piano practice. Due to the difficulty of collecting a large amount of individual's performance data to learn, we adopt the Bayesian inference, which has the advantage that it can be used even if only a small amount of learning data is available. Thus, we propose a method for treating gaze (EHS) probabilistically and integrating gaze and keying information within the Bayesian inference framework. First, we assume the length of EHS follows the normal distribution and that the distribution is updated dynamically by observed data. We define the distribution of keying information in the same way. Next, we integrate gaze and keying information, using Gaussian Mixture Model, to be used as the likelihood function in the Bayesian inference. Advantages of the method include improvement of the accuracy of estimating current keying position and the ease of adding other new features which reflect the user's individuality and/or thinking.

## 2. RELATED WORK

### 2.1 Gaze Information and Individuality

A performer perceives music while forming 'chunks', which are units to recognize a sequence of pitch events as a pattern. The size of EHS is related to the size of a chunk. Weaver revealed that professional performers perceive the

Copyright: ©2019 Kaede Noto et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



notes on a score as horizontal and vertical groupings [20]. Furneaux et al. conducted experiments to identify the differences in EHS between professional and amateur pianists [7]. In this research, EHS was defined as the number of notes between the note(s) being played and the note(s) upon which the player's gaze was fixed (performance point and gaze point). The professional's EHS (approx. four notes) was larger than the amateur's (approx. two notes). From the point of view of melody, Kobori and Takahashi compared the eye movements when professionals and amateurs play a melody on piano and guitar [9]. This research suggested that the individuality of performers, the difficulty of music pieces and the performer's knowledge on music pieces were the crucial factors which influenced eye movement.

## 2.2 Automatic Score Following

A central issue in score following is correctly estimating performance position in response to a variety of uncertain events such as leaping forward, mis-keying and repeating. Nakamura et al. developed a score-following system, Eurydice, that estimated the user's current position using improved HMM and Viterbi algorithm [17]. Since the weight was calculated based on distance, Eurydice tended to estimate a position near the previous current position. To some extent, Eurydice achieved accurate estimation in performance containing unexpected movements such as mistakes, repeats, and skips. However, since Eurydice used only keying information, it was difficult to identify the phrase being played in the case of a melody containing many repeated phrases.

Terasaki et al. proposed a score-following system that was hardly affected by unexpected movements [13]. The system introduced gaze likelihood to the cost of DP matching. Gaze likelihood was obtained by HMM, which was employed for predicting gaze and removing noise from the raw data of eye movement. The system could correctly estimate current position, even when a player started at a point different from the previous point at which he/she had stopped playing, or a beginning point of a repeated phrase. Terasaki et al. evaluated the estimation error rate for musical scores including repeated phrases and found that the correct answer rate was improved by 1.2 times (from 72.7% to 85.2%). However, the system unfortunately could not cope with the individuality of EHS.

Grubb et al. introduced the Bayesian inference into an automatic accompaniment system [6]. They defined the probability distribution with respect to note number  $i$  as a random variable, which was updated every time data was observed. Parameter  $d$  stands for the estimated distance from a pre-estimated position,  $v$  the observation value, and  $j$  the performer's position at the previous timing. The estimated position was updated by the most recent observation. First, based on the previous position and estimated distance, the current position is estimated, as follows.

$$f_{I|D}(i|d) = \int_{j=0}^{\|Score\|} f_{I-J|D}(i-j|d) \cdot f_{Source}(j) \partial j$$

Next, this estimated value is further updated to take into

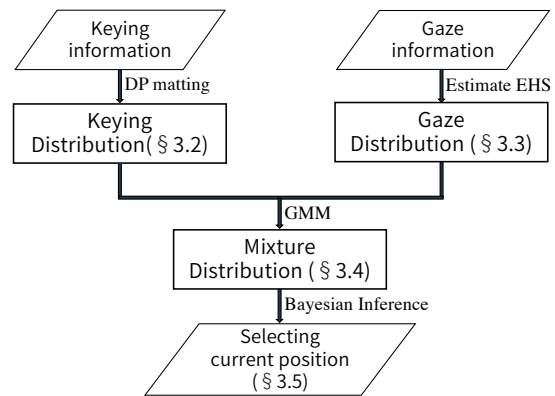


Figure 1. System Configuration

account the observation using the Bayesian inference, as follows.

$$f_{I|D,V}(i|d, v) \propto f_{V|I}(v|i) \cdot f_{I|D}(i|d)$$

$f_{V|I}(v|i)$  is made from observation, and is considered as a likelihood. In an experiment on recorded data and real-time data, a latency of 159 ms occurred on average. This value falls below the limit of latency that humans can perceive, which is 10~100 ms [5]. Thus, Grubb's system did not interrupt piano performance, but users reported that they felt some discomfort.

## 3. A SCORE-FOLLOWING SYSTEM USING KEYING AND GAZE INFORMATION

This section describes the score-following model which deals with gaze, and shows how the input data are converted to distributions and how they are combined.

### 3.1 Combining Gaze information with Keying Information

Fig.1 shows the processing flow of the proposed system. The model takes two input data at the same time: keying information and gaze information. The keying information as MIDI numbers and the gaze information measured by an eye-tracking device are entered into the system. The system fits the input data to normal distributions, and integrates the distributions using GMM. After integrating the distributions, by Bayesian inference, the system estimates a note number on a score as current position ( $i \in \{1, 2, 3, \dots, I\}$ ).

### 3.2 Distribution of Keying

To formalize the keying information as a normal distribution, we need to define the average and the variance. Firstly, to obtain the average, we use DP matching. DP matching finds the degree of similarity between notes on a score and notes being played, called the best match, and chooses the score position having the best match as a performance position [12]. However, it is difficult for DP

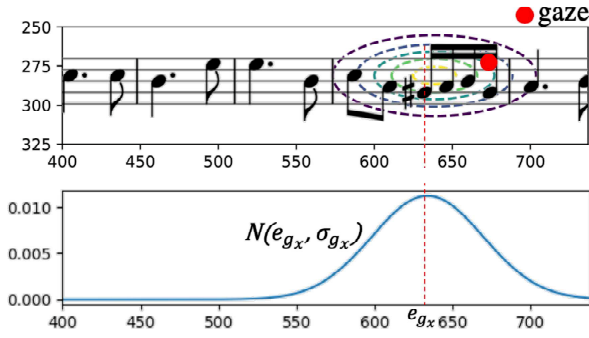


Figure 2. Create the distribution of gaze, with note estimated from EHS as  $\mu$

matching to follow a performance which includes backward leaps and repeats. As a simple solution, we employ an exhaustive search of all the played notes over an entire score, as in our previous research [13]. Then, we can identify the note number  $e_{DP} \in \{1, 2, 3, \dots, I\}$  that is most probably the average value. Next, we define a value corresponding to the variance. For all notes, we calculate the average distance  $\sum |x_i - x_{i+1}|/I$  between every adjacent note in the horizontal direction. Let us regard the value obtained as the standard deviation of the distribution of keying. Then, we define the keying distribution as follows:

$$p_{DP}(i) \sim N(\mu_{DP}, \sigma_{DP})$$

where  $\mu_{DP}$  is equal to  $e_{DP}$  and  $\sigma_{DP}$  is  $(\sum |x_i - x_{i+1}|/I)^2$ .

### 3.3 Distribution of Gaze

To introduce the gaze information, we define the distribution of gaze considering eye-hand-span (EHS) in Fig.2.

EHS consists of the horizontal and vertical spans  $x_i - g_x$  and  $y_i - g_y$ , where  $(x_i, y_i)$  means the note number of the key being played and  $(g_x, g_y)$  the gaze point on a display (where, on the display, the user is looking). Here, concerning the size of EHS, we assume it follows the normal distribution:

$$p_x(x_i - g_x) \sim \mathcal{N}(x_i - g_x | \mu_{g_x}, \sigma_{g_x}) \quad (1)$$

$$p_y(i_y - g_y) \sim \mathcal{N}(i_y - g_y | \mu_{g_y}, \sigma_{g_y}) \quad (2)$$

Terms  $x_i - g_x$  and  $x_i - g_y$  are the lengths of EHS in the horizontal and vertical directions, respectively. Variables  $\mu_g$  and  $\sigma_g$  represent the average length of EHS and the variance, respectively. Since the Gauss-gamma distribution, as prior distribution, is used in the Bayesian inference,  $\mu_g$  and  $\sigma_g$  can be analytically calculated.

According to the length of EHS learned, we estimate the user's current position from gaze point  $(g_x, g_y)$  and assign  $g_x$  and  $g_y$  to Equations(1) and (2). Since, at this moment, more than one candidate note is obtained, to determine the user's current position, we choose a note by calculating the likelihood of each note, considering EHS, as follows:

$$e_g = \arg \max_{i \in I} [p_x(i) p_y(i)]$$

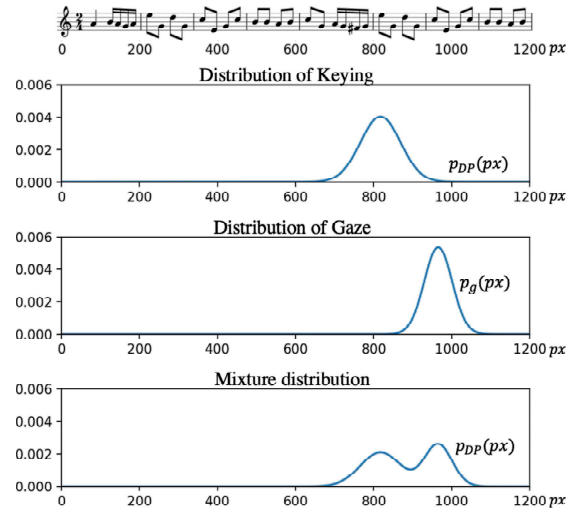


Figure 3. Integration of distributions by GMM

Then, using the above variables, we define the gaze distribution,  $p_g(i)$ , with  $e_g$  as the average and  $\sigma_{g_x}$  as the variance (Fig.2).

$$p_g(i) \sim N(i | e_{g_x}, \sigma_{g_x})$$

### 3.4 Integration of Multiple Information

We use GMM to integrate the keying and gaze distributions (Fig.3). To use GMM, we convert the random variable of keying distribution into a coordinate value on the x axis on a score (in units of pixels). The integrated probability distribution by GMM is given by Equation(3) with mixture ratio  $\pi_k$ :

$$P_{GMM}(i) = \sum_{k=1}^2 \pi_k N(i | \mu_k, \sigma_k) \quad (3)$$

$$\sum_{k=1}^K \pi_k = 1$$

where  $\mu_k = [e_{DP}, e_g]$  and  $\sigma_k = [\sigma_{DP}, \sigma_{g_x}]$ .

Since we need to decide the significance of each type of information, we use the EM algorithm to update mixture ratio  $\pi_k$ .

$$r(Z_{nk}) = \frac{\pi_k N(x | \mu_k, \sigma_k)}{\sum_{i=1}^2 \pi_i N(x | \mu_i, \sigma_i)}$$

$$\pi_k = \frac{\sum_{n=1}^N r(Z_{nk})}{N}$$

Here,  $x$  represents an observation, which is assumed to have been sampled from the normal distribution of average  $\mu_i$  and variance of  $\sigma_i$ . To estimate  $\pi_k$ , we calculate the  $r(Z_{nk})$  which represents the ratio of the distribution of each  $k$  ( $k = 1, 2$ ) at the values of the density function of the mixed distribution at pixel  $x$ . Since we cannot know what a player is thinking, in principle, we cannot know the true point at which he/she is playing.

To obtain as accurate a value of  $\pi_k$  as possible, we instructed subjects to play notes in the order indicated by a score, that is, linearly from the beginning to the end.

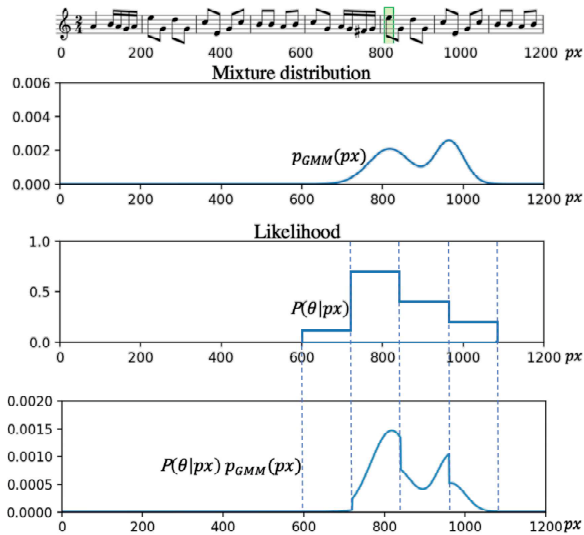


Figure 4. Estimating Note by Multiplication of Likelihood and Mixture Distribution

Mixture ratio  $\pi_k$  represents the maximum likelihood of the mixing ratio. In E step, we calculate the expected value of  $P_{GMM}$  when the mixture ratio is  $\pi_k$ . In M step,  $\pi_k$  is optimized by the data obtained by sampling for each note from a normal distribution that assumes  $\mu = i_x$ ,  $\sigma_g = (\sum |x_i - x_{i+1}|/I)^2$  by the maximum likelihood estimation. From the obtained mixture ratio, we define the mixed distribution that is the weighted summation of the keying and gaze information (Fig.3).

### 3.5 Estimating Keying Position based on Bayesian Inference

The current position is estimated from the mixture distribution and likelihood. The relationship between a note and the variables can be represented by the occurrence probability of the  $i$ -th note,  $P(i)$ , and the value distribution of variables,  $P(\theta)$ . To estimate  $P(i|\theta)$  using Bayesian estimation, it is necessary to know  $P(i)$  and  $P(\theta|i)$  in advance.

However, it is difficult to uniquely determine  $P(i)$  because a performance includes errors and leaps forward and backward. Thus, we substitute mixed distribution for  $P(i)$ . Accordingly, we estimate  $P(\theta|px)$ , where  $px$  means the discrete frequency distribution of  $\theta$  when note  $i$  occurs. The parameter  $\theta$  represents a random variable about the combination of gaze and keying information.

Thus, the random variables depend on the combination of note numbers and gaze points. To prevent  $\theta$  from being sparse, the number of random variables was limited by using a gaze range, whereby the musical score was divided into 10 parts in the horizontal direction, instead of gaze points. The bottom part of Fig.4 shows the result of multiplying the mixture distribution by the likelihood. The current performance point  $e_m$  is determined so that the multiplication of  $P(\theta|i)$  and  $P_{GMM}(i)$  is maximized.

$$e_m = \arg \max_i P(\theta|i) P_{GMM}(i)$$

## 4. EXPERIMENT

### 4.1 Experimental Description

By properly assigning the parameters, which are gaze distribution, keying distribution and mixture rate, our system can estimate a user's current position correctly. The parameters are drawn from ground truth data which consists of true current position, MIDI key numbers, and gaze points. First, by MIDI data we know current notes that subjects are playing including wrong key strokes. Even if a subject played a wrong key, a subjects are instructed to continue playing without trying to recover wrong key strokes during piano performance as if a subject plays correct notes. Then we align the note that a subject plays with the corresponding gaze data. Thereby, the system acquires gaze distribution, keying distribution, and mixture ratio for each individual subject. After the parameters are identified, we start the experiment.

### 4.2 Implementation of Proposed Method

If we take a set piece that the subjects already know, it is possible that EHS will be biased due to prior knowledge of the phrases or decreased score reading time. Therefore, we should adopt a piece that none of the subjects knows. We select a piece from Yamaha Music Ability Test (Grade 5 Grade) Sight playing / Improvisation and extract the right-hand part of 27 measures (103 notes) [1]. The set piece contains three identical phrases and there are nine duplicated notes (Fig. 5). On a screen, we show the score image, which is made with MuseScore to remove any musical symbols such as staccato and slur. Thus, the bar lengths of a score on a screen are variable depending on the number of notes and symbols in each bar.

The GUI of the system is implemented in Processing and the model part in Python. The system obtains keying information from the MIDI keyboard and gaze information from a gaze measuring device, Eye-Tribe ET1000, which does not disturb performance because it is small and stationary [15]. The effective range of the distance between the device and user's eyes is 45 to 75cm, and the spatial resolution is the angle of 0.1 degree, which means the resolution of 0.17cm, 50cm ahead. To calibrate the system, sixteen points are showed on a screen one by one. The frame rate of the Eye-Tribe generates sampling data of gaze information at a frame rate of 30 Hz.

The gaze data is transmitted to the model part in Open Sound Control (OSC), which is the protocol for communication among computers, sound synthesizers, and other multimedia devices. After estimating a user's current position by the algorithm described previously, the model part transmits it to GUI. To superimpose the position of the estimated playing note, the melody of the set piece is displayed in the x-axis of note number  $i$  and the y-axis of MIDI note number.

### 4.3 Evaluation Procedure

To obtain experimental data, we asked seven subjects (Subjects A to G) to play a set piece. Five of the subjects (A to E) had experience of learning the piano, and the other two



Figure 5. The set piece that contains three identical phrases and there are nine duplicated notes

(F and G) did not but could read a musical score. If a subject plays a melody linearly following a score, of course, the system can estimate almost every note correctly. Next, subjects were instructed to play parts of the piece in reverse order. For instance, subjects played from the 20th to the 23rd measure immediately before they played from the 6th to the 9th measure (Fig. 5). Here, the accuracy rate was defined as the rate at which performance position is correctly estimated with respect to all keystrokes.

#### 4.4 Results and Discussion

Table 1 shows the number of passed-over keys, that of mistakenly pressed keys, the temporal interval between the restart of the system and the timing of capturing correct position (latency in the table), and the accuracy rate. In regard to the accuracy rate, we see a large gap between two groups: a group that has a rate of more than 90% (Subjects A to D and F) and the other group (E and G). It does not seem that the accuracy rates are related to the numbers of passed-over and mistakenly pressed keys. Concerning these numbers, Subjects F and G reach the largest numbers.

Table 2 shows the weights of the mixture rate. For Subjects A to E who have experience of learning the piano, the weights of DP matching are larger than those of the gaze distribution. In contrast, for Subjects F and G (non-experienced), the weights of gaze are larger than those of DP matching. We think a reason for this is that GMM determines the weights by the maximum likelihood. Hence, during score following, keying information plays an im-

Table 1. Accuracy rate and uncertain factors

	Subject	passed-over	missed	latency	acc. (%)
experienced	A	1	0	1	93.3
	B	0	0	2	93.3
	C	0	0	2	93.3
	D	0	0	0	90.0
	E	0	0	1	70.0
non experienced	F	1	2	0	96.6
	G	0	2	-	0.0

Table 2. The Weight of Mixture Rate

	Subject	weight of Gaze	weight of DP
experienced	A	0.13	0.87
	B	0.13	0.87
	C	0.15	0.85
	D	0.26	0.74
	E	0.24	0.76
non experienced	F	0.48	0.52
	G	0.76	0.24

Table 3. The Differences Between Gaze Points Before and After the Experiment

	Subject	Gap for x-axis(px)	Gap for y-axis(px)
experienced	A	1.9	-0.7
	B	29.8	-1.6
	C	-56.0	-70.9
	D	-27.0	-104.7
	E	-12.9	19.0
non-experienced	F	23.2	-143.7
	G	-85.8	-146.4

portant role for the experienced subjects, while gaze information is important for the non-experienced ones.

To examine the performance of the eye-tracking device, we instructed the subjects to look at the same points on the score before and after the experiment. Table 3 shows the average differences of gaze points before and after the experiment, called Gap in the table. The table shows that the absolute values of the experienced subject's gaps are smaller. The absolute value of G's gap is the largest, being shifted left by 85.8 px and upward by 146.4 px. These values correspond to a shift of a half measure to the left and almost 1 staff up in Fig. 5.

While the experimental results show that the proposed method determines the weights of information, to which the user's individuality is adapted, with little data, there are some cases in which position cannot be identified. To examine such cases, let us consider the relationship between



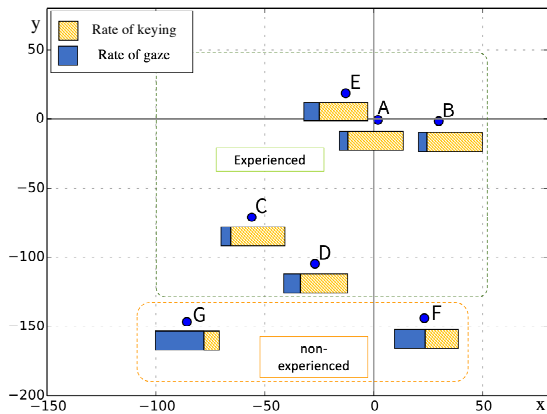


Figure 6. Relationship Between the Gap and the Mixture Rate

misalignment and the mixture rate. Fig. 6 shows the relationships between the gaps and the mixture rates in the scatter plot of the gap data in Table 3. The bars under dots represent the mixture rates for gaze and keying distributions. In the figure, we can see a trend in which the accuracy depends on the mixture ratio. In particular, the gaze mixture rate of Subject F is 0.48, on the other hand, that of Subject G is 0.76. Although the gaze information of subject G is weighted more heavily, G's gaps are also large. We think it is for this reason that the accuracy rate decreases.

## 5. CONCLUSION

In this paper, we proposed a score-following system which adapts a user's individuality in piano performance. We fit gaze information and keying information to the normal distribution and integrate them into the Bayesian inference by using GMM. The experimental results demonstrate the relevance of each type of information to the user's individuality. In particular, the keying information is important for an experienced performer, on the other hand, the gaze information is important for a non-experienced one. However, regarding the accuracy of the system, large differences occur among subjects. We think one of the reasons is not considering misalignment of the eye-tracking system. Future work will include developing a robust method to deal with errors related to eye-tracking, and adding other kinds of information which reflect user's individuality and/or mind, such as gesture and blinking.

## Acknowledgments

This work has been supported by JSPS Kakenhi 16H01744 and 16K12560, and JST CREST Grant Numbers JPMJCR17A1 and JPMJCR17A3, Japan.

## 6. REFERENCES

- [1] Y. Asano, "Piano Performance Grade 5 Sight playing, Improvisation Variant Example Song Collection ('90 Revised Edition)", Yamaha Music, 1997.
- [2] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment", ICMC, pp.193-198, 1984.

- [3] R. B. Dannenberg, and H. Mukaino, "New techniques for enhanced quality of computer accompaniment", ICMC, 1988.
- [4] M. Dorfer, A. Arzt, and G. Widmer, "Towards score following in sheet music images", ISMIR, 2016.
- [5] M. Dorfer, F. Henkel, and G. Widmer, "Learning to Listen, Read, and Follow: Score Following as a Reinforcement Learning Game", ISMIR, 2018.
- [6] P. Desain, H. Honing, "Tempo Curves Considered Harmful. In Music Mind and Machine", Amsterdam: Thesis Publishers, pp.25-40, 1992.
- [7] S. Furneaux, M. F. Land, "The Effects of Skill on the Eye-Hand Span During Musical Sight-Reading", Proceedings of the Royal Society of London B, 266, pp. 2435-2440.
- [8] L. Grubb, R. B. Dannenberg, "Enhanced Vocal Performance Tracking Using Multiple Information Sources", ICMC, pp.37-44, 1998.
- [9] S. Kobori, K. Takahashi, "Cognitive Processes During Piano and Guitar Performance: An Eye Movement Study", ICMPC, pp.748-751, 2008.
- [10] C. Oshima, K. Nishimoto and M. Suzuki, "A piano duo performance support system to motivate children's practice at home (in Japanese)", IPSJ, 46(1), pp. 157-171, 2005.
- [11] S. Sagayama, T. Nakamura, E. Nakamura, Y. Saito, H. Kameoka, and N. Ono, "Automatic Music Accompaniment Allowing Errors and Arbitrary Repeats and Jumps", POMA, vol. 21, 035003, pp. 1-11, 2014.
- [12] H. Takeda, T. Nishimoto, and S. Sagayama, "Automatic Accompaniment System of MIDI Performance Using HMM-based Score Following (in Japanese)", IPSJ, 2006.
- [13] S. Terasaki, Y. Takegawa, K. Hirata, "Proposal of Score-Following Reflecting Gaze Information on Cost of DP matching", ICMC, pp. 144-149, 2017.
- [14] J. Sloboda, "The eye - hand span - an approach to the study of sight reading", Psychology of Music, 2(2), pp. 4-10, 1974.
- [15] The Eye Tribe, available from <https://theeyetribe.com/> [25 Dec2018].
- [16] E. Nakamura, P. Cuvillier, A. Cont, N. Ono, and S. Sagayama, "Autoregressive Hidden Semi-Markov Model of Symbolic Music Performance For Score Following", ISMIR, 2015.
- [17] E. Nakamura, T. Nakamura, Y. Saito, N. Ono, and S. Sagayama, "Outer-product hidden Markov model and polyphonic MIDI score following", Journal of New Music Research, vol. 43, no. 2, pp. 183-201, Apr 2014.
- [18] B. Pardo and W. Birmingham, "Modeling form for on-line Following of Musical Performances", Proc. of the Twentieth National Conference on Artificial Intelligence, 2005.
- [19] V. Thomas, C. Fremerey, M. Muller, and M. Clausen, "Linking Sheet Music and Audio Challenges and New Approaches", Dagstuhl Follow-Ups, 3: pp. 1-22, 2012.
- [20] Weaver, H. E. , "Studies of Ocular Behavior in Music Reading", Psychological Mono-graphs, 55(1), pp.1-29, 1943.

# Alternative Measures: A Musicologist Workbench for Popular Music

**Beach Clark**

Georgia Institute of Technology  
Bclark66@gatech.edu

**Claire Arthur**

Georgia Institute of Technology  
Claire.arthur@gatech.edu

## ABSTRACT

The objective of this project is to create a digital “workbench” for quantitative analysis of popular music. The workbench is a collection of tools and data that allow for efficient and effective analysis of popular music. This project integrates software from pre-existing analytical tools including music21 but adds methods for collecting data about popular music. The workbench includes tools that allow analysts to compare data from multiple sources. Our working prototype of the workbench contains several novel analytical tools which have the potential to generate new musicological insights through the combination of various datasets. This paper demonstrates some of the currently available tools as well as several sample analyses and features computed from this data that support trend analysis. A future release of the workbench will include a user-friendly UI for non-programmers.

## 1. INTRODUCTION

One of the challenges to the scholarly analysis of popular music is the difficulty of collecting symbolic data. There are two forms of symbolic musical data: symbolic representations of the music itself (i.e., score-based or MIDI data), and symbolic *metadata*. Metadata can refer to relatively large-scale features *about* a song as a whole (e.g., title, artist, track length, etc.), or may refer to *features* computed from the data such as chord progressions, key estimates, number of sections, etcetera. Since both the raw audio and any published scores for virtually all popular music fall under copyright protection, musicologists wishing to study popular music are largely restricted to the analysis of symbolic metadata. In order to follow trends in the rapidly evolving field of popular music, large volumes of symbolic metadata will have to be curated through automated or semi-automated approaches. Standardized analytical metrics also need to be developed to complement and enhance qualitative analyses. This project focused on three objectives: collecting symbolic musical metadata from multiple sources; integrating newly developed analysis tools with existing tools; and developing a hierarchical model of metrics that can help guide this type of analysis.

### 1.1 Collecting musical data from multiple sources

Musical analysis is a complex process that involves tasks such as transcribing lyrical, harmonic, rhythmic and melodic information, and doing research into the provenance of the music (i.e. who wrote the lyrics, who wrote the music, who

produced the track, etc.). Many computational analysis projects begin with manual collection and analysis of the data, a process that is immensely time consuming. Due to the resources required in curating datasets, many analyses are carried out on relatively small samples (e.g., [19, 25]). However, an additional challenge in *any* analysis is in collecting enough data to be able to make statistically valid inferences about a larger population. Using data that has already been collected can substantially improve the time and resources invested for a given study by allowing analysts to focus on analyzing data instead of collecting it. To use this data effectively, however, multiple sources must be combined, the validity of the data must be tested, and the quality or usefulness of that data further refined. In part, this project investigates the efficacy of using data from midi collections combined with data from commercial sources like Spotify, as well as websites like Ultimate Guitar, to carry out musicological analyses.

The Spotify database contains features and metadata for approximately forty million songs that is continuously updated with new material based on listener tastes. This makes Spotify and similar sites potentially excellent sources for structural data such as song length and number of sections. This project developed a method of collecting and storing data from the Spotify API that makes the data easier to use for musical analysis.

While music listening websites and software (e.g., Spotify, LastFM) are good sources of structural data about music, some musical analysis tasks involve making or using transcriptions of lyrical and harmonic information which are not easily obtained from these sources. Websites like e-Chords [24] and Ultimate Guitar [23], however, contain user-submitted chord transcriptions of popular songs for thousands of songs online. Although the quality of the transcriptions varies, some scholars have found that using these transcriptions in conjunction with other symbolic data can be used to improve the accuracy of automated chord transcription and key detection [15, 17]. This project developed a web scraper for Chordify.net that gathers chord information for selected songs. We tested the validity of this extracted chord information against two known dataset of “expert” chord transcriptions for the same songs [5, 19]. Our results showed that, for certain tasks, such as measuring distributions of

chord usage, the differences between the Chordify transcriptions and the experts' transcriptions were not significant.<sup>1</sup> We are currently using Chordify chord transcriptions in an analysis that shows the impact of "harmonic surprise" on listener perceptions.

## 1.2 Integrating newly developed and existing tools

Once the data curation step is completed, it still has to be analyzed. There are a number of existing tools currently available for symbolic musical analysis, namely music21 and humdrum. Music21 is a Python-based, open source toolkit which provides a needed bridge between the demands of music scholars and of computer researchers [4]. It has an active support community and provides support for a number of analysis tasks such as Roman Numeral Analysis, or metrical analysis. For users who are familiar with music21, taking advantage of existing functionality speeds the analysis process. This project developed a converter to allow symbolic metadata from Spotify to be parsed into music21's native format.

## 1.3 Post-analysis of extracted features

All of the data sources investigated by this project have been used independently by other researchers (e.g., Dieleman [2], Gauvin [29], Thomas [27]). The present authors recognize that there is value in developing automated methods for feature extraction from this collective data that can be used to systematically analyze large samples, and leverage these methods in the current project.

This metadata may not be appropriate for every analysis task. For example, the pitch vectors from Spotify do not have octave information and result from the integration of multiple voices, making them difficult to use for melodic analysis. For other tasks, such as analysis of form, the same metadata may offer a significant increase in the amount of music to be analyzed than could be covered with other methods. In some cases, however, additional processing of the metadata will be required, and scholars will have to adjust their methodologies to take best advantage of this type of data.

# 2. RELATED WORK

## 2.1. Data collection and feature extraction

Similar data collections to those described above have been used in previous work. In 2011, Bertin-Mahieux and Ellis (Columbia University) along with Lamere and Whitman (EchoNest) created the Million Song Dataset (MSD) to address the issue of the lack of data that can be used to analyze popular music [1]. The dataset consisted of a collection of precomputed features extracted from audio along with metadata from one million popular songs. The project also included code to retrieve audio samples of some of the songs from 7digital. Dieleman, Brakel and Schrauwen used the dataset (in particular Echonest pitch vectors and timbre vectors) to create machine learning models for key detection, artist recognition and genre detection [2]. The MSD has been used

by several other scholars (e.g. Gallay [26], Thomas [27]) since its creation, highlighting the value of being able to use pre-collected data from a very large dataset. All features in the MSD were extracted by Echonest. Unfortunately, for popular music scholars, the MSD has not been updated since its creation in 2011. In 2015, Spotify acquired Echonest, creating a continuously updated source for this type of data. To make the metadata more accessible to music scholars, this project created an SQL database to make the Spotify data easier to search and combine with data from other sources.

In 2016, Raffel collected over 178,000 MIDI transcriptions of complete popular songs to support his doctoral research [3]. A somewhat smaller but more rigorous dataset – the McGill Billboard dataset was developed at McGill University [5]. This dataset is a collection of transcriptions of selected songs from the Billboard Hot 100 for the period 1955 – 1991. In total it contains the annotations and audio features corresponding to 890 of the entries from the random sample of *Billboard* chart slots as presented at ISMIR 2011. In 2010, McVicar and De Bie used chord transcriptions from the e-chords website to show that using chord transcriptions from publicly available web resources can improve the accuracy of Hidden Markov models using chroma features from audio [15].

These four projects used collection methods that have important differences. The pitch and timbre vectors were computed from audio using digital signal processing techniques. The MIDI transcriptions were obtained using web scraping software to crawl numerous public MIDI sites. The transcriptions in the McGill Billboard dataset were hand-transcribed by music scholars. The e-chord website is a collection of chord transcriptions that are contributed by users who have varying levels of musical training. Web scraping software was also used to collect the chord transcriptions.

Each type of data has proven valuable to music scholars. Various types of pitch vectors like the ones in Spotify have been used for music information retrieval tasks such as key detection [2, 7] and chord estimation. MIDI transcriptions have been used for tasks like comparing musical sequences and chord detection [6, 3]. Pitch, timbre, beat and section vectors along with chord transcriptions from web resources have been used to compile historical analyses of trends in popular music [16].

A workbench tool most similar to the present project was developed by Abdallah et. al, called the Digital Music Lab (DML) system, which is a large collection of metadata containing both low-level features and collection-level analyses that is stored in a carefully planned architecture [30]. The DML system contains a rich selection of features similar to those found in the MSD and has a wider selection of genres. The primary differences between the current project and the DML system is that our project is compatible with symbolic music, and allows the user to curate their own dataset from theoretically *any* existing song or work, whereas with the DML the user is limited to songs in the DML system, which is comparatively lacking in popular music data.

Popular music scholars need easy-to-use methods to collect relevant data about a specific collection. They also need to

<sup>1</sup> This comparison was made by taking the distribution of the counts of all simplified RN hand analyses and comparing against those computed by Chordify.

be able to search the data and visualize it quickly across multiple dimensions without having to develop programming skills to do so. The traditional approach has been to analyze scores, which show the data in a concise and easy to use format. This project proposes additional formats for the data.

### 3. PROJECT OVERVIEW

This project focuses on three objectives. The first is to identify efficient and effective methods of collecting and curating symbolic data to support musical analysis, including MIDI data collected from multiple sources, hand transcriptions of scores, data from music listening sites like Spotify and chord transcriptions from websites similar to eChord.

The second is to investigate approaches for integrating existing analysis tools with newly developed software.

The third is to demonstrate the concept of combining data from multiple sources by building several datasets to perform representative analyses for musicological tasks.

#### 3.1 Data Collection Methods

As a demonstration of the collection methods in our workbench, we describe the assembly of a subset of musical data for analysis. We chose to examine the weekly list of the “Billboard Hot 100” for the period of 1980 – 1989. A web scraper for the Billboard website was built to retrieve the list and a SQLite3 table was created to enable it to be searched. Data from Spotify for each of the songs was collected using the Spotify API.

Based on the Spotify API documentation, the primary focus of exposing the API is to allow developers to create applications to enhance Spotify premium listeners’ experience by recommending songs and playlists from the Spotify collection. Unfortunately for musical analysis, the API search method only supports full text searches. Specifically, filtering parameters are limited to: artist, title, genre, year, market, upc and sirc (<https://developer.spotify.com/documentation/web-api/reference/search/search/>). Even though the returned data contains large amounts of fine-grained information, it needs further data manipulation to be used in analysis tasks (e.g., section lengths are identified, but need to be grouped and compared according to section *type*—verse, chorus, etc.) To make the data easier to search and filter for the user, we created a new SQLite3 database. Using this database, analysts can now select finer-grained musical features (e.g., section length, number of sections, highest ranking) for specific songs from either source (Spotify or Billboard) in a single location.

An examination of our database schema reveals a number of attributes that can be useful for popular music analysis. For example, our “track\_feature” table has metadata elements extracted from Spotify that describe timbral characteristics of a song, such as liveness, acousticness, speechiness and energy; rhythmic characteristics, such as danceability; and mood characteristics, such as valence (see [28] for definitions of these features). We define additional tables, including: “track\_section”, “track\_bar”, “track\_beat,” and “track\_tatum”, that contain metadata describing structural and metrical characteristics. The “track\_section” table also

contains the estimated key and mode (major or minor) of a section. The “billboard\_tracks” table has the artist, label, and title of the song, and the weekly ranking (1 – 100) pulled from the Billboard API. Using our new database, it is relatively straightforward to search for specific songs and compute, for example, the average duration, number of sections or standard deviation of the duration of the beats in the song. This allows analysts to answer questions like “is there a trend toward shorter songs in the Billboard Hot 100 during the past 20 years?” or, “Is there a historical trend towards music being more “in the pocket?” (i.e. tightly aligned to a specific pulse without drifting over the course of the song). As a prototype, we also developed an Excel workbook containing the Billboard 1980’s song data that allows analysts without programming skills to analyze and visualize the data.<sup>1</sup>

#### 3.2 Integration with Existing Tools

Popular music analysts will continue to need more specialized tools for sophisticated musical tasks such as metrical or Roman Numeral analysis. This project developed a converter to allow symbolic data from Spotify to be parsed into music21. The converter parses pitch, beat onset, duration, tempo, time signature, and bar vectors to create a music21 stream. The resulting stream is a hierarchical representation of the structural features of a song (i.e., section, bars, beats) linked to their timing information derived from the audio. Once the stream has been created, music21 tools can be used to analyze key, quantize or transpose the symbolic data so that it can be compared to other songs. For example, if pitch vectors were retrieved into a music21 stream and segmented by musical event, as was done in [21], our database could be used with music21 to work on various symbolic MIR problems such as chord estimation—something we are, in fact, currently testing.

#### 3.3 Analysis Examples

In order to validate the efficacy of the approaches to building datasets and using new and existing tools to analyze data, our project developed several comparative analyses. These are described in the following sections.

##### 3.3.1 Large, surface-level features

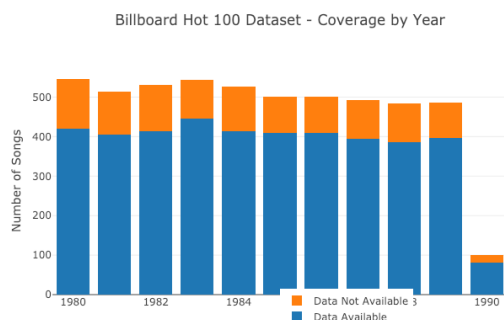
One reason for building a database of musical data is to ensure that the inferences made from the analyses are statistically valid for some larger population. As mentioned above, previous studies have mentioned the challenges of defining a representative musical population [19]. Here we demonstrate the utility of collecting large amounts of musical data to augment statistical power. We queried the Billboard “Hot 100” for the list of songs on the chart for the period of 1980 through 1989. We then collected feature data from Spotify using our database. For the 10-year period, Figure 1 shows the Billboard Hot 100 had 4,226 total songs of which 3,554 were available on Spotify (79.4%).

As an exploratory exercise, we analyzed several descriptive features extracted from this database for the same set of songs. The following paragraphs describe these sample features. Figure 2 shows a histogram of the distribution of songs produced by various labels for the period 1980 – 1989. The

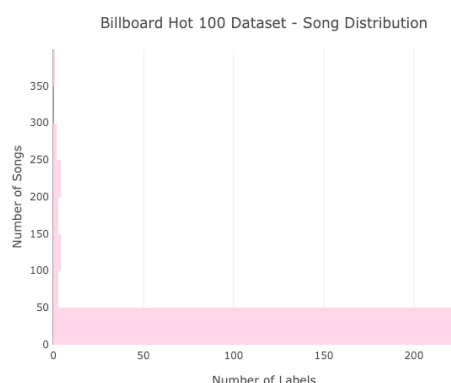
<sup>1</sup> The workbook used in this analysis is in the github repository at <https://github.com/bclark288/alternative-measures>



distribution shows that 225 labels produced less than 100 songs each while two labels (Columbia and Atlantic) produced between 240 and 400 songs each. This finding raises additional questions such as whether the same labels continue to dominate in later years and whether there are similarities between the rhythm, pitch, timbre and other musical information for a label. In other words, do labels have a “style”?



**Figure 1.** Coverage of songs from the weekly Billboard Hot 100 in the Spotify database for the period 1980 – 1989.



**Figure 2.** Distribution of Songs Produced by Label.

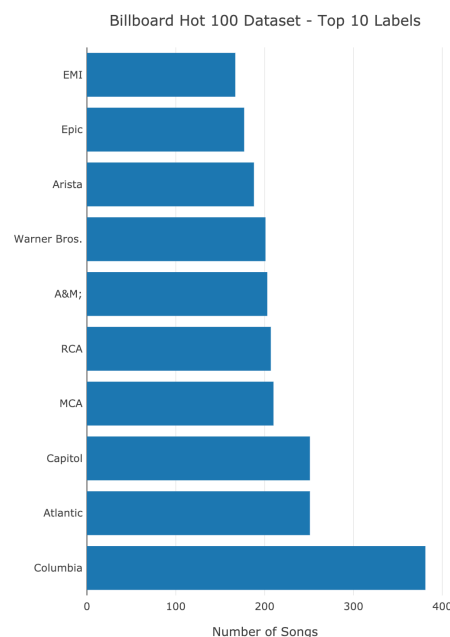
Figure 3 shows the detail of the number of songs produced by the top 10 labels. Additional questions to be considered are, whether the number of songs on the charts are simply a function of the size of the label, or whether some smaller labels are able to produce better rankings. Answering these questions might give insights into how much of an effect the marketing resources of a label affect the popularity of a recording.

Figure 4 shows the number of songs produced by the top 10 artists for the same period. This figure shows a more even distribution of songs among artists. Additional questions suggested by this analysis are whether artists with popular recordings in a given timeframe (i.e. weeks or months) show similarities in musical features.

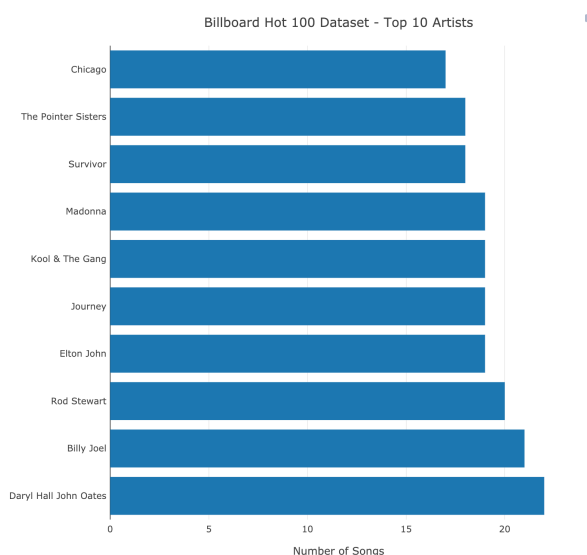
Figure 5 shows the average length and number of sections of songs for the period. A section is described in the Spotify developer guide as a portion of a song that shows a signifi-

cant change in rhythm or timbre [20]. The number of sections according to the Spotify data is higher than would be expected if one equated the word “section” with “verse” or “chorus”.

Both features are relatively stable on average for the entire period, but questions to be answered might be whether the length and type of sections vary when chart rank is considered. Another question might be whether the tempo of a group of songs varies from one section to another, and if so, how much. One unexpected result of this analysis was that

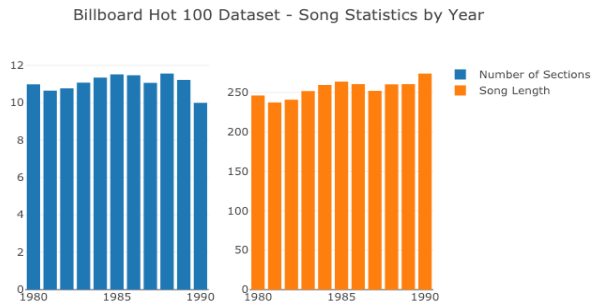


**Figure 3.** Detail of Songs Produced by the Top 10 Labels 1980 – 1989.



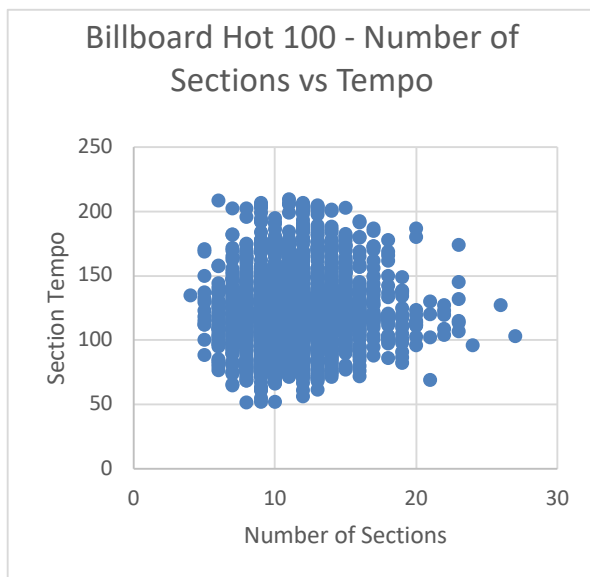
**Figure 4.** Songs Produced by the Top 10 Artists 1980 – 1989.

on average, the songs in this time period were nearly 4 minutes long.



**Figure 5.** Average Number of Sections and Average Song Length.

Figure 6 shows a scatter plot of the number of sections vs. section tempo for each of the songs for the period. The number of sections seems to cluster at between 7 and 20 and the tempo clusters between 50 and 200. The project compared section lengths of selected songs from this data with descriptions made by a human annotator [22]. The section boundaries did not match exactly. However, they were similar enough to warrant further investigation into how song structure computed from audio features differs with that of human annotators. Another interesting question raised by this data is how often the tempo of sections changes within a given song.



**Figure 6.** Section Tempo vs Number of Sections for Billboard Hot 100 songs from 1980 – 1989.

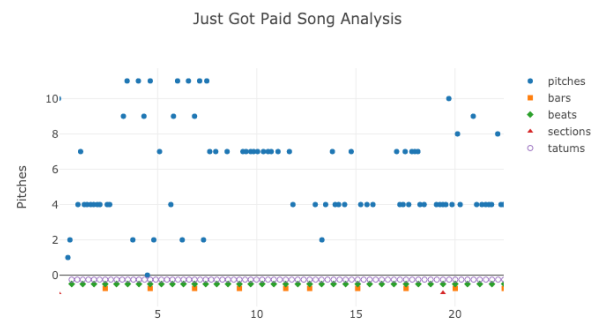
Figure 7 shows a portion of an Excel Pivot Table that shows the modality, tempo and section boundaries for a selection of songs by Bruce Springsteen. High level structural analytics such as the examples above can be used to identify songs that have unusual structural characteristics for a deeper analysis.

Figure 8 shows a portion of a detailed breakdown of the metric structure of “Just Got Paid” by ZZ Top. The graph shows the relationship between pitch events, bars, beats (the

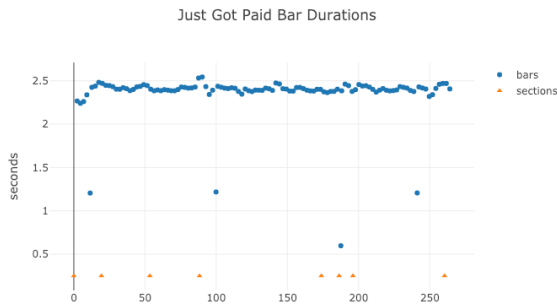
numerator of the time signature) and tatums (defined as the lowest regular pulse train that a listener intuitively infers from the timing of perceived musical events) [20]. Figure 9 shows a breakdown of the length of each bar in seconds for “Just Got Paid”. This visualization not only shows the variability in tempo, but that there is a detectable pattern. As mentioned above, this type of detailed analysis can be helpful in troubleshooting large scale analysis systems. It can also be useful to help conceptualize which low-level audio features contribute to a given style.

Average of tempo		Column Labels	
Row Labels		0	1
Bruce Springsteen		131.4411095	125.4315651
Born In The USA			108.8407
0			120.841
26.23255			120.885
57.9783			120.902
145.30752			120.937
177.05396			120.931
184.99255			120.878
208.80727			120.917
241.07735			121.152
255.94716			120.964
281.52897			0
Brilliant Disguise			126.4764545
Cover Me		120.9813333	120.6081667
Dancing In The Dark		147.618	147.603
Fade Away		105.119	105.3159
Fire		134.7676667	135.137
Glory Days		114.788	118.4452222
Hungry Heart		109.87825	110.1416667
I'm Goin' Down		132.94	132.43625
I'm On Fire		139.4036667	139.506
My Hometown			118.036
One Step Up		150.089	150.293625
Tunnel Of Love			117.073
War		120.07	120.059125
Madonna		116.4961409	122.7400299

**Figure 7.** Excel Pivot table of Modality and Tempo of Selected Song Sections. Numbers in the left column under the heading “Born in the USA” are start times of each section. The two columns on the right are the tempo of the section. If the tempo is under the heading “0”, the section is minor. “1” is major.

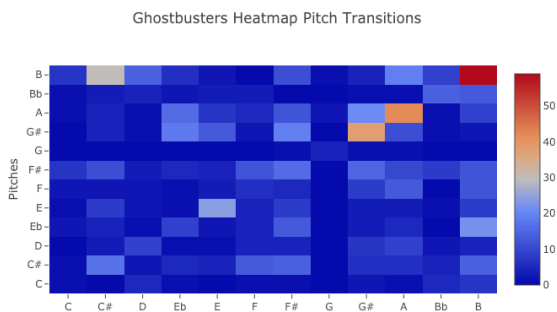


**Figure 8.** Detailed Song Analysis of one section of “Just Got Paid”.



**Figure 9.** Graph of bar durations from “Just Got Paid”.

Figure 10 shows a heatmap with pitch class transitions from the song “Ghostbusters”. The pitch data from Spotify has proven to be more difficult to use in large scale analysis. However, as mentioned above, researchers have used chroma vectors similar to the Spotify chroma vectors with additional preprocessing to predict harmony. As an experiment, this project converted the pitch information from Spotify for “Little Sister” to midi format using music21 and then played the resulting stream in GarageBand. Based on listening to the output, it is unlikely that this data can be used for melodic analysis without significant filtering.



**Figure 10.** Pitch Transitions heatmap from “Ghostbusters”.

### 3.3.2 Chord detection

Learning chord progressions is an important skill in analyzing popular music. As a result, websites like Ultimate Guitar and eChords have become popular resources for informal music education. However, given the level of effort and experience required to transcribe the harmonic content of popular songs, there has been considerable motivation to develop automated chord transcription algorithms [18]. These algorithms typically are applied to raw audio data. However, since we are largely trying to evaluate the utility of the symbolic data (e.g., pitch vectors) output from Spotify, we investigated four approaches to using existing tools and data to perform chord estimation. Code for the models is in a github repository at <https://github.com/bclark288/alternative-measures>.

<sup>1</sup> Accuracy was determined by dividing the total number of correct chord labels produced by the model by the total number of labels in the ground truth dataset).

Music21 has key finding algorithms based on the Krumhansl-Schmuckler algorithm. Research by Delgado and Naples suggests that this key finding algorithm can be adapted to the task of chord recognition [6]. As a prototype, we developed a tool to parse Spotify song data (as described in section 3.2 above), analyze the harmonic content, and compare the results of the analysis to ground truth harmonic analysis data (McGill Billboard project). The overall accuracy for the prototype was 11.6%.<sup>1</sup> Although the overall accuracy of this approach was low, the errors showed a tendency to be relatively close to the ground truth harmonically.

Research suggests that using pattern matching in combination with Hidden Markov models (HMM) can yield good results in chord estimation. In collaboration with Nestor Naples, this project modified a pattern-matching/HMM model to accept pitch vectors from Spotify in place of the chroma vectors computed from NNLS. Results for this model were also disappointing (significantly less than 50%, using the same accuracy measure as identified above). Possible reasons for the weak results were related to the difficulty of computing segments from the pitch class vectors and aligning them with the timing of the ground truth data.

In an effort to improve the accuracy of chord detection using pitch and timbre vectors, a convolutional recurrent neural network was developed. We tested this using a dataset of chroma and timbre vectors for 890 popular songs from the McGill Billboard dataset.

Accuracy using this model was considerably better (over 50% training accuracy using the measure described above), although still not at standard performance accuracy for this type of algorithm. The poor results suggest at least two factors are inhibiting the usefulness of precomputed pitch vectors for harmonic analysis: the challenge of aligning segments with ground truth labels, and a need for additional filtering or preprocessing of the pitch vectors.

In sum, since we do not have access to the audio, nor to the algorithms that produced the pitch vector data, using this aggregated data for harmonic analysis remains a complex computational problem. However, we are continuing to evaluate other possible solutions, as the success of a model such as this one would be of high value.

## 4. CONCLUSIONS

### 4.1 Spotify database and data collection

This project demonstrated the utility of using symbolic data computed from audio and extracted from the Spotify database for certain musical analysis tasks such as comparing songs on the basis of length, number of sections and modality of sections. Our goal is to implement a method that relies on data extracted from a large, continuously updated data source (i.e., Spotify) in combination with other sources so that researchers may be able to examine musical questions via a larger and more appropriate sample of data. By creating new tools to extract novel features, samples can be easily analyzed to see how they compare (e.g. how do popular songs from 1980 – 1989 compare to songs from the past year).

While we demonstrated the utility of the workbench for relatively large-scale or surface features, the disappointing results of our attempts at chord recognition shows that the data may not be suited for every task.

Although this project focused on developing tools to collect data from Spotify, work by other researchers also demonstrated the efficacy of collecting data from sources like eChords and the sites that house midi transcriptions of popular songs. These sites also give access to lyrical content—a musical feature that is grossly understudied [23].

Additionally, sites like Spotify and Soundcloud give invaluable insights into audience perception. Review sites (e.g., genius.com) can provide valuable information relating to important musical features as well as audience perception information. Thus, we feel that the ideal dataset for analysis of popular music will contain data from multiple sources.

#### 4.2 Integration with existing tools

Our sample analyses demonstrated the value of analyzing data from different sources using a toolkit with ready-to-use routines by developing a converter for music21 that parses Spotify data into a music21 stream. Computational analysis toolkits such as music21 have useful components that perform melodic, rhythmic and harmonic analysis. By integrating our workbench with these existing tools, we take advantage of their existing functionality.

At present, we are currently developing a web scraping tool to collect chord information from Ultimate Guitar.com and Chordify.net. Given that other researchers have used chord transcriptions from websites like e-chords.com [15], future work will need to include integrating the prototype with existing tools such as music21, etc. As popular music scholars develop more quantitative analyses of popular music, extensions to the existing tools as well as novel tools and features will need to be developed. A priority for future improvements to our workbench is to include friendly user interfaces to allow some of the tasks to be performed by people without any software development or programming skills.

#### 4.3 Comparative analysis

Finally, our project demonstrated the value of performing comparative musical analysis by combining data across multiple sources (e.g., finding Billboard hits within the Spotify dataset). We evaluated the usability of the data itself as well as developed new features for comparison. Given the complexity of some types of analysis (e.g., harmonic analysis), more difficult tasks will require substantial manual intervention. However, we aim to include several automated tools like the ones shown here to handle routine tasks, which will ultimately improve both the quality and timeliness of an analyst's work.

### 5. FUTURE WORK

While this project focused on proof-of-concept prototypes to demonstrate the value of updating/creating musicology tools to make them easier to use in the study of popular music, additional work is needed in the following areas:

#### 5.1 User Interface

The prototypes developed in this project are still mostly dependent on the users having python programming skills. The excel workbook (e.g., see Figure 12) developed to perform some of the example comparative analyses is an example of the type of tool that can be used to improve an analyst's workflow. Since there are many existing popular tools for data visualization, (e.g. Sharepoint/PowerBI, Tableau, Plotly), future work will include such tools as well as developing an online version of the workbench featuring user-friendly interfaces for non-programmers.

#### 5.2 Chord detection

The three initial prototypes for chord detection from Spotify pitch class profiles show that more work is needed. However, given the importance of harmony in popular music, development of this feature will be valuable. Other research in this area that suggests that pitch class vectors similar to those output from Spotify (i.e. EchoNest pitch class profiles) can be used for key or chord detection [2], and, that chord detection may be improved with some additional processing [7]. One key challenge that will have to be addressed in future work is the alignment of the time segments in the Spotify data with the segments in the ground truth dataset [13].

#### 5.3 Development of new metadata

In this paper, our analyses made use of computed metadata such as average song length, average section length, and section modality to aid the analysis of popular music and to discover features that define musical style. Future work will create frameworks for more complex schema that, for example, could deal with the analysis at the intersection of multiple features (e.g., form and harmonic content). This will dramatically facilitate the cross-comparison of multiple features such as chord progressions, rhythmic patterns and timbre to identify elements of style.

### REFERENCES

- [1] T. Bertin-Mahieux, D. Ellis, B. Whitman and P. Lamere. "The Million Song Dataset," 12<sup>th</sup> International Society for Music Information Retrieval Conference (ISMIR 2011).
- [2] S. Dieleman, P. Brakel and B. Schrauwen. "Audio-Based Music Classification with a Pre-Trained Convolutional Network," 12<sup>th</sup> International Society for Music Information Retrieval Conference (ISMIR 2011).
- [3] C. Raffel. "Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching". Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Graduate School of Arts and Sciences Columbia University 2016
- [4] M. Cuthbert, C. Ariza, L. Friedland. "Feature Extraction and Machine Learning On Symbolic Music Using the



- music21 Toolkit". 12<sup>th</sup> International Society for Music Information Retrieval Conference (ISMIR 2011).
- [5] J. Burgoyne, J. Wild, and I. Fujinaga. "An Expert Ground Truth Set for Audio Chord Recognition and Music Analysis". 12th International Society for Music Information Retrieval Conference (ISMIR 2011).
  - [6] N. Delgado and N. Napoles. "Symbolic Chord Detection". Final Project of the Music Information Retrieval seminar taught by Emilia Gómez as part of the Master in Sound and Music Computing 2016-2017, <https://zenodo.org/record/1888500#.XAVori-ZPOS>
  - [7] S. Kamonsantiroj, L. Wannatrong and L. Pipanmaekaporn. "Improving Pitch Class Profile for Musical Chords Recognition Combining Major Chord Filters and Convolution Neural Networks," 2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), Hamamatsu, Shizuoka, Japan, 2018, pp. 880-885.
  - [8] J. Covach. "Popular Music, Unpopular Musicology," in N. Cook and M. Everist, Eds., *Rethinking Music* (Oxford University Press, 1999), 452 – 70.
  - [9] R. Middleton. "Popular Music Analysis and Musicology: Bridging the Gap," *Popular Music*, Vol 12, No. 2 (May, 1993), pp 177 – 190.
  - [10] M. Katz: "What does it mean to study popular music?". A Musicologist's perspective," *Journal of Popular Music Studies*, Volume 26, Issue 1, Pages 22 – 27.
  - [11] D. Huron: "The Humdrum User Guide". Available Online: [www.humdrum.org](http://www.humdrum.org) (accessed on 3 January 2019).
  - [12] Trevor Owen DeClercq. "Sections and Successions in Successful Songs: A Prototype Approach to Form in Rock Music." Ph. D. Dissertation University of Rochester, Rochester, NY, 2012.
  - [13] B. Pardo and W. Birmingham. "Algorithms for Chordal Analysis," *Computer Music Journal*, 26:2, pp 27-49, Summer 2002.
  - [14] D. Etherington. Spotify Acquires the Echo Nest Gaining Control of The Music DNA That Powers Its Rivals. 2015. Available Online: <https://techcrunch.com/2014/03/06/spotify-acquires-the-echo-nest/> (accessed on 7 January 2019).
  - [15] M. McVicar and T. De Bie. "Enhancing Chord Recognition Accuracy Using Web Resources," *Proceedings of 3<sup>rd</sup> International Workshop on Machine Learning and Music*, pp 41-44, 2010.
  - [16] M. Mauch, R. MacCallum, M. Levy and A. Leroi. "The Evolution of Popular Music: USA 1960 – 2010," *Royal Society of Open Science* 2.150081
  - [17] M. Mauch, K. Noland, S. Dixon. "Using Musical Structure to Enhance Automatic Chord Transcription," 10<sup>th</sup> International Society for Music Information Retrieval Conference (ISMIR 2009).
  - [18] E. Humphrey, J. Bello. "Four Timely Insights on Automatic Chord Estimation," 16<sup>th</sup> International Society for Music Information Retrieval Conference (ISMIR 2015).
  - [19] T. DeClercq, D. Temperly. "A Corpus Analysis of Rock Harmony," *Popular Music* (2011) Volume 30/1, pp 47-70.
  - [20] Spotify Developers Manual. Available Online: <https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-analysis/> (accessed on 21 December 2018).
  - [21] N. Napoles, "Chordal\_analysis\_music21". Available Online: [https://github.com/napulen/chordal\\_analysis\\_music21](https://github.com/napulen/chordal_analysis_music21) (accessed on 27 December 2018).
  - [22] J. Covach. *What's That Sound? An Introduction to Rock and It's History*, 2<sup>nd</sup> Edition W. W. Norton Company, Inc.: 500 5<sup>th</sup> Avenue, New York, NY. 2006; pp 444 – 491.
  - [23] Ultimate Guitar. Available Online. <https://www.ultimate-guitar.com> (accessed on 14 December 2018).
  - [24] E-Chords.com. Available Online. <https://www.e-chords.com> (accessed on 12 December 2018).
  - [25] Summach, J. (2011). "The structure, function, and genesis of the prechorus". *Music Theory Online*, 17(3).
  - [26] Million Song Project. Available Online: <https://cgallay.github.io/Ada/> (accessed on 29 December 2018).
  - [27] Million Song Dataset Analysis. Available Online: <https://github.com/thomasSve/Million-Song-Dataset-Analysis> (accessed on 14 January 2019).
  - [28] Spotify Developers Manual. Available Online: <https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/> (accessed on 21 December 2018).
  - [29] Gauvin, H. "The Times They Were a Changin'": A Database-Driven Approach to the Evolution of Harmonic Syntax in Popular Music from the 1960's," *Empirical Musicology Review*, Volume 10 2015.
  - [30] S Abdallah, E Benetos, N Gold, S Hargreaves, T Weyde, D Wolff, "The digital music lab: a big data infrastructure for digital musicology" *Journal on Computing and Cultural Heritage (JOCCH)* 10 (1), 2 2017.

# Autoencoders for music sound modeling: a comparison of linear, shallow, deep, recurrent and variational models

Fanny Roche<sup>1,3</sup>   Thomas Hueber<sup>3</sup>   Samuel Limier<sup>1</sup>   Laurent Girin<sup>2,3</sup>

<sup>1</sup>Arturia, Meylan, France   <sup>2</sup>Inria Grenoble Rhône-Alpes, France

<sup>3</sup>Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, Grenoble, France

fanny.roche@gipsa-lab.fr

## ABSTRACT

This study investigates the use of non-linear unsupervised dimensionality reduction techniques to compress a music dataset into a low-dimensional representation which can be used in turn for the synthesis of new sounds. We systematically compare (shallow) autoencoders (AEs), deep autoencoders (DAEs), recurrent autoencoders (with Long Short-Term Memory cells – LSTM-AEs) and variational autoencoders (VAEs) with principal component analysis (PCA) for representing the high-resolution short-term magnitude spectrum of a large and dense dataset of music notes into a lower-dimensional vector (and then convert it back to a magnitude spectrum used for sound resynthesis). Our experiments were conducted on the publicly available multi-instrument and multi-pitch database NSynth. Interestingly and contrary to the recent literature on image processing, we can show that PCA systematically outperforms shallow AE. Only deep and recurrent architectures (DAEs and LSTM-AEs) lead to a lower reconstruction error. The optimization criterion in VAEs being the sum of the reconstruction error and a regularization term, it naturally leads to a lower reconstruction accuracy than DAEs but we show that VAEs are still able to outperform PCA while providing a low-dimensional latent space with nice “usability” properties. We also provide corresponding objective measures of perceptual audio quality (PEMO-Q scores), which generally correlate well with the reconstruction error.

## 1. INTRODUCTION

Deep neural networks, and in particular those trained in an unsupervised (or self-supervised) way such as autoencoders [1] or GANs [2], have shown nice properties to extract latent representations from large and complex datasets. Such latent representations can be sampled to generate new data. These types of models are currently widely used for image and video generation [3–5]. In the context of a project aiming at designing a music sound synthesizer driven by high-level control parameters and propelled by data-driven machine learning, we investigate the use of such techniques for music sound generation as an alternative to classical music sound synthesis techniques like

additive synthesis, subtractive synthesis, frequency modulation, wavetable synthesis or physical modeling [6].

So far, only a few studies in audio processing have been proposed in this line, with a general principle that is similar to image synthesis/transformation: projection of the signal space into a low-dimensional latent space (encoding or embedding), modification of the latent coefficients, and inverse transformation of the modified latent coefficients into the original signal space (decoding).

In [7, 8], the authors implemented this principle with autoencoders to process normalized magnitude spectra. An autoencoder (AE) is a specific type of artificial neural network (ANN) architecture which is trained to reconstruct the input at the output layer, after passing through the latent space. Evaluation was made by computing the mean squared error (MSE) between the original and the reconstructed magnitude spectra.

In [9], NSynth, an audio synthesis method based on a time-domain autoencoder inspired from the WaveNet speech synthesizer [10] was proposed. The authors investigated the use of this model to find a high-level latent space well-suited for interpolation between instruments. Their autoencoder is conditioned on pitch and is fed with raw audio from their large-scale multi-instrument and multi-pitch database (the NSynth dataset). This approach led to promising results but has a high computational cost.

Another technique to synthesize data using deep learning is the so-called variational autoencoder (VAE) originally proposed in [11], which is now popular for image generation. A VAE can be seen as a probabilistic/generative version of an AE. Importantly, in a VAE, a prior can be placed on the distribution of the latent variables, so that they are well suited for the control of the generation of new data. This has been recently exploited for the modeling and transformation of speech signals [12, 13] and also for music sounds synthesis [14], incorporating some fitting of the latent space with a perceptual timbre space. VAEs have also been recently used for speech enhancement [15–17].

In line with the above-presented studies, the goal of the present paper is i) to provide an extensive comparison of several autoencoder architectures including shallow, deep, recurrent and variational autoencoders, with a systematic comparison to a linear dimensionality reduction technique, in the present case Principal Component Analysis (PCA) (to the best of our knowledge, such comparison of non-linear approaches with a linear one has never been done in previous studies). This is done using both an objec-

Copyright: © 2019 Roche et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

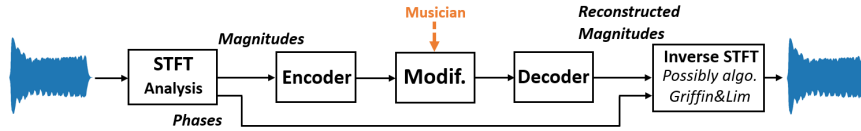


Figure 1: Global diagram of the sound analysis-transformation-synthesis process.

tive physical measure (root mean squared error – RMSE) and an objective perceptual measure (PEMO-Q [18]); ii) to compare the properties of the latent space in terms of correlation between the extracted dimensions; and iii) to illustrate how interpolation in the latent space can be performed to create interesting hybrid sounds.

## 2. METHODOLOGY

The global methodology applied for (V)AE-based analysis-transformation-synthesis of audio signals in this study is in line with previous works [7, 8, 12, 13]. It is illustrated in Fig. 1 and is described in the next subsections.

### 2.1 Analysis-Synthesis

First, a Short-Term Fourier Transform (STFT) analysis is performed on the input audio signal. The magnitude spectra are sent to the model (encoder input) on a frame-by-frame basis, and the phase spectra are stored for the synthesis stage. After possible modifications of the extracted latent variables (at the bottleneck layer output, see next subsection), the output magnitude spectra is provided by the decoder. The output audio signal is synthesized by combining the decoded magnitude spectra with the phase spectra, and by applying inverse STFT with overlap-add. If the latent coefficients are not modified in between encoding and decoding, the decoded magnitude spectra are close to the original ones and the original phase spectra can be directly used for good quality waveform reconstruction. If the latent coefficients are modified so that the decoded magnitude spectra become different from the original one, then the Griffin & Lim algorithm [19] is used to estimate/refine the phase spectra (the original phase spectra are used for initialization) and finally reconstruct the time-domain signal. A few more technical details regarding data pre-processing are given in Section 3.2.

### 2.2 Dimensionality Reduction Techniques

**Principal Component Analysis:** As a baseline, we investigated the use of PCA to reduce the dimensionality of the input vector  $\mathbf{x}$ . PCA is the optimal linear orthogonal transformation that provides a new coordinate system (i.e. the latent space) in which basis vectors follow modes of greatest variance in the original data [20].

**Autoencoder:** An AE is a specific kind of ANN traditionally used for dimensionality reduction thanks to its diabolical shape [21], see Fig. 2. It is composed of an encoder and a decoder. The encoder maps a high-dimensional low-level input vector  $\mathbf{x}$  into a low-dimensional higher-level latent vector  $\mathbf{z}$ , which is assumed to nicely encode properties or

attributes of  $\mathbf{x}$ . Similarly, the decoder reconstructs an estimate  $\hat{\mathbf{x}}$  of the input vector  $\mathbf{x}$  from the latent vector  $\mathbf{z}$ . The model is written as:

$$\mathbf{z} = f_{\text{enc}}(\mathbf{W}_{\text{enc}}\mathbf{x} + \mathbf{b}_{\text{enc}}) \quad \text{and} \quad \hat{\mathbf{x}} = f_{\text{dec}}(\mathbf{W}_{\text{dec}}\mathbf{z} + \mathbf{b}_{\text{dec}}),$$

where  $f_{\text{enc}}$  and  $f_{\text{dec}}$  are (entry-wise) non-linear activation functions,  $\mathbf{W}_{\text{enc}}$  and  $\mathbf{W}_{\text{dec}}$  are weight matrices and  $\mathbf{b}_{\text{enc}}$  and  $\mathbf{b}_{\text{dec}}$  are bias vectors. For regression tasks (such as the one considered in this study), a linear activation function is generally used for the output layer.

At training time, the weight matrices and the bias vectors are learned by minimizing some cost function over a training dataset. Here we consider the mean squared error (MSE) between the input  $\mathbf{x}$  and the output  $\hat{\mathbf{x}}$ .

The model can be extended by adding hidden layers in both the encoder and decoder to create a so-called deep autoencoder (DAE), as illustrated in Fig. 2. This kind of architecture can be trained globally (end-to-end) or layer-by-layer by considering the DAE as a stack of shallow AEs [1, 22].

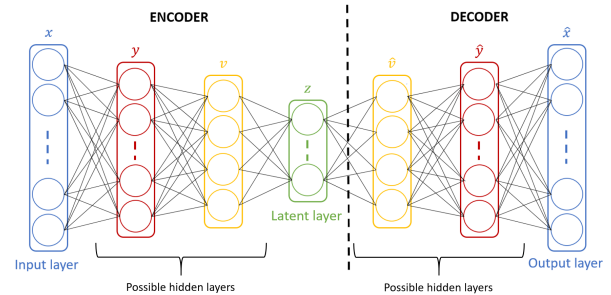


Figure 2: General architecture of a (deep) autoencoder.

**LSTM Autoencoder:** In a general manner, a recurrent neural network (RNN) is an ANN where the output of a given hidden layer does not depend only on the output of the previous layer (as in a feedforward architecture) but also on the internal state of the network. Such internal state can be defined as the output of each hidden neuron when processing the previous input observations. They are thus well-suited to process time series of data and capture their time dependencies. Such networks are here expected to extract latent representations that encode some aspects of the sound dynamics. Among different existing RNN architectures, in this study we used the Long Short-Term Memory (LSTM) network [23], which is known to tackle correctly the so-called vanishing gradient problem in RNNs [24]. The structure of the model depicted in Fig. 2 still holds while replacing the classical neuronal cells by LSTM cells, leading to a LSTM-AE. The cost function to optimize remains the same, i.e. the MSE between the input  $\mathbf{x}$  and the

output  $\hat{\mathbf{x}}$ . However, the model is much more complex and has more parameters to train [23].

**Variational Autoencoder:** A VAE can be seen as a probabilistic AE which delivers a parametric model of the data distribution, such as:

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z}),$$

where  $\theta$  denotes the set of distribution parameters. In the present context, the likelihood function  $p_\theta(\mathbf{x}|\mathbf{z})$  plays the role of a probabilistic decoder which models how the generation of observed data  $\mathbf{x}$  is conditioned on the latent data  $\mathbf{z}$ . The prior distribution  $p_\theta(\mathbf{z})$  is used to structure (or regularize) the latent space. Typically a standard Gaussian distribution  $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$  is used, where  $\mathbf{I}$  is the identity matrix [11]. This encourages the latent coefficients to be mutually orthogonal and lie on a similar range. Such properties may be of potential interest for using the extracted latent coefficients as control parameters of a music sound generator. The likelihood  $p_\theta(\mathbf{x}|\mathbf{z})$  is defined as a Gaussian density:

$$p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\theta(\mathbf{z}), \boldsymbol{\sigma}_\theta^2(\mathbf{z})),$$

where  $\boldsymbol{\mu}_\theta(\mathbf{z})$  and  $\boldsymbol{\sigma}_\theta^2(\mathbf{z})$  are the outputs of the decoder network (hence  $\theta = \{\mathbf{W}_{\text{dec}}, \mathbf{b}_{\text{dec}}\}$ ). Note that  $\boldsymbol{\sigma}_\theta^2(\mathbf{z})$  indifferently denotes the covariance matrix of the distribution, which is assumed diagonal, or the vector of its diagonal entries.

The exact posterior distribution  $p_\theta(\mathbf{z}|\mathbf{x})$  corresponding to the above model is intractable. It is approximated with a tractable parametric model  $q_\phi(\mathbf{z}|\mathbf{x})$  that will play the role of the corresponding probabilistic encoder. This model generally has a form similar to the decoder:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \tilde{\boldsymbol{\mu}}_\phi(\mathbf{x}), \tilde{\boldsymbol{\sigma}}_\phi^2(\mathbf{x})),$$

where  $\tilde{\boldsymbol{\mu}}_\phi(\mathbf{x})$  and  $\tilde{\boldsymbol{\sigma}}_\phi^2(\mathbf{x})$  are the outputs of the encoder ANN (the parameter set  $\phi$  is composed of  $\mathbf{W}_{\text{enc}}$  and  $\mathbf{b}_{\text{enc}}$ ;  $\tilde{\boldsymbol{\sigma}}_\phi^2(\mathbf{x})$  is a diagonal covariance matrix or is the vector of its diagonal entries).

Training of the VAE model, i.e. estimation of  $\theta$  and  $\phi$ , is done by maximizing the marginal log-likelihood  $\log p_\theta(\mathbf{x})$  over a large training dataset of vectors  $\mathbf{x}$ . It can be shown that the marginal log-likelihood can be written as [11]:

$$\log p_\theta(\mathbf{x}) = \text{D}_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})|p_\theta(\mathbf{z}|\mathbf{x})) + \mathcal{L}(\phi, \theta, \mathbf{x}),$$

where  $\text{D}_{\text{KL}} \geq 0$  denotes the Kullback-Leibler divergence (KLD) and  $\mathcal{L}(\phi, \theta, \mathbf{x})$  is the variational lower bound (VLB) given by:

$$\mathcal{L}(\phi, \theta, \mathbf{x}) = \underbrace{-\text{D}_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})|p_\theta(\mathbf{z}))}_{\text{regularization}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{reconstruction accuracy}}. \quad (1)$$

In practice, the model is trained by maximizing  $\mathcal{L}(\phi, \theta, \mathbf{x})$  over the training dataset with respect to parameters  $\phi$  and  $\theta$ . We can see that the VLB is the sum of two terms. The first term acts as a regularizer encouraging the approximate posterior  $q_\phi(\mathbf{z}|\mathbf{x})$  to be close to the prior  $p_\theta(\mathbf{z})$ . The second term represents the average reconstruction accuracy. Since the expectation w.r.t.  $q_\phi(\mathbf{z}|\mathbf{x})$  is difficult to compute analytically, it is approximated using a Monte Carlo estimate

and samples drawn from  $q_\phi(\mathbf{z}|\mathbf{x})$ . For other technical details that are not relevant here, the reader is referred to [11].

As discussed in [12] and [25], a weighting factor, denoted  $\beta$ , can be introduced in (1) to balance the regularization and reconstruction terms:

$$\mathcal{L}(\phi, \theta, \beta, \mathbf{x}) = -\beta \text{D}_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})|p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})], \quad (2)$$

This enables the user to better control the trade-off between output signal quality and compactness/orthogonality of the latent coefficients  $\mathbf{z}$ . Indeed, if the reconstruction term is too strong relatively to the regularization term, then the distribution of the latent space will be poorly constrained by the prior  $p_\theta(\mathbf{z})$ , turning the VAE into an AE. Conversely, if it is too weak, then the model may focus too much on constraining the latent coefficients to follow the prior distribution while providing poor signal reconstruction [25]. In the present work we used this type of  $\beta$ -VAE and we present the results obtained with different values of  $\beta$ . These latter were selected manually after pilot experiments to ensure that the values of the regularization and the reconstruction accuracy terms in (2) are in the same range.

### 3. EXPERIMENTS

#### 3.1 Dataset

In this study, we used the NSynth dataset introduced in [9]. This is a large database (more than 30 GB) of 4s long monophonic music sounds sampled at 16 kHz. They represent 1,006 different instruments generating notes with different pitches (from MIDI 21 to 108) and different velocities (5 different levels from 25 to 127). To generate these samples different methods were used: Some acoustic and electronic instruments were recorded and some others were synthesized. The dataset is labeled with: i) instrument family (e.g., keyboard, guitar, synth\_lead, reed), ii) source (acoustic, electronic or synthetic), iii) instrument index within the instrument family, iv) pitch value, and v) velocity value. Some other labels qualitatively describe the samples, e.g. brightness or distortion, but they were not used in our work.

To train our models, we used a subset of 10,000 different sounds randomly chosen from this NSynth database, representing all families of instruments, different pitches and different velocities. We split this dataset into a training set (80%) and testing set (20%). During the training phase, 20% of the training set was kept for validation. In order to have a statistically robust evaluation, a  $k$ -fold cross-validation procedure with  $k = 5$  was used to train and test all different models (we divided the dataset into 5 folds, used 4 of them for training and the remaining one for test, and repeated this procedure 5 times so that each sound of the initial dataset was used once for testing).

#### 3.2 Data Pre-Processing

For magnitude and phase short-term spectra extraction, we applied a 1,024-point STFT to the input signal using a sliding Hamming window with 50% overlap. Frames corre-



sponding to silence segments were removed. The corresponding 513-point positive-frequency magnitude spectra were then converted to log-scale and normalized in energy: We fixed the maximum of each log-spectrum input vector to 0 dB (the energy coefficient was stored to be used for signal reconstruction). Then, the log-spectra were thresholded, i.e. every log-magnitude below a fixed threshold was set to the threshold value. Finally they were normalized between  $-1$  and  $1$ , which is a usual procedure for ANN inputs. Three threshold values were tested:  $-80$  dB,  $-90$  dB and  $-100$  dB. Corresponding denormalization, log-to-linear conversion and energy equalization were applied after the decoder, before signal reconstruction with transmitted phases and inverse STFT with overlap-add.

### 3.3 Autoencoder Implementations

We tried different types of autoencoders: AE, DAE, LSTM-AE and VAE. For all the models we investigated several values for the encoding dimension, i.e. the size of the bottleneck layer / latent variable vector, from  $enc = 4$  to  $100$  (with a fine-grained sampling for  $enc \leq 16$ ). Different architectures were tested for the DAEs:  $[513, 128, enc, 128, 513]$ ,  $[513, 256, enc, 256, 513]$  and  $[513, 256, 128, enc, 128, 256, 513]$ . Concerning the LSTM-AE, our implementation used two vanilla forward LSTM layers (one for the encoder and one for the decoder) with non-linear activation functions giving the following architecture:  $[513, enc, 513]$ . Both LSTM layers were designed for many-to-many sequence learning, meaning that a sequence of inputs, i.e. of spectral magnitude vectors, is encoded into a sequence of latent vectors of same temporal size and then decoded back to a sequence of reconstructed spectral magnitude vectors. The architecture we used for the VAE was  $[513, 128, enc, 128, 513]$  and we tested different values of the weight factor  $\beta$ . For all the neural models, we tested different pairs of activation functions for the hidden layers and output layer, respectively: (tanh, linear), (sigmoid, linear) and (tanh, sigmoid).

AE, DAE, LSTM-AE and VAE models were implemented using the *Keras* toolkit [26] (we used the *scikit-learn* [27] toolkit for the PCA). Training was performed using the Adam optimizer [28] with a learning rate of  $10^{-3}$  over 600 epochs with early stopping criterion (with a patience of 30 epochs) and a batch size of 512. The DAEs were trained in two different ways, with and without layer-wise training.

### 3.4 Experimental Results for Analysis-Resynthesis

Fig. 3 shows the reconstruction error (RMSE in dB) obtained with PCA, AE, DAE and LSTM-AE models on the test set (averaged over the 5 folds of the cross-validation procedure), as a function of the dimension of the latent space. The results obtained with the VAE (using the same protocol, and for different  $\beta$  values) are shown in Fig. 4. For the sake of clarity, we present here only the results obtained for i) a threshold of  $-100$  dB applied on the log-spectra, and ii) a restricted set of the tested AE, DAE and VAE architectures (listed in the legends of the figures). Similar trends were observed for other thresholds and other tested architectures. For each considered dimension of the

latent space, a 95% confidence interval of each reconstruction error was obtained by conducting paired t-test, considering each sound (i.e. each audio file) of the test set as an independent sample.

RMSE provides a global measure of magnitude spectra reconstruction but can be insufficiently correlated to perception depending on which spectral components are correctly or poorly reconstructed. To address this classical issue in audio processing, we also calculated objective measures of perceptual audio quality, namely PEMO-Q scores [18]. The results are reported in Fig. 5 and Fig. 6.

As expected, the RMSE decreases with the dimension of the latent space for all methods. Interestingly, PCA systematically outperforms (or at worst equals) shallow AE. This somehow contradicts recent studies on image compression for which a better reconstruction is obtained with AE compared to PCA [1]. To confirm this unexpected result, we replicated our PCA vs. AE experiment on the MNIST image dataset [29], using the same AE implementation and a standard image preprocessing (i.e. vectorization of each  $28 \times 28$  pixels gray-scale image into a 784-dimensional feature vector). In accordance with the literature, the best performance was systematically obtained with AE (for any considered dimension of the latent space). This difference of AE's behavior when considering audio and image data was unexpected and, to our knowledge, it has never been reported in the literature.

Then, contrary to (shallow) AE, DAEs systematically outperform PCA (and thus AE), with up to almost 20% improvement (for  $enc = 12$  and  $enc = 16$ ). Our experiments did not reveal notable benefit of layer-by-layer DAE training over end-to-end training. Importantly, for small dimensions of the latent space (e.g. smaller than 16), RMSE obtained with DAE decreases much faster than with PCA and AE. This is even more the case for LSTM-AE which shows an improvement of the reconstruction error of more than 23% over PCA (for  $enc = 12$  and  $enc = 16$ ). These results confirm the benefits of using a more complex architecture than shallow AE, here deep or recurrent, to efficiently extract high-level abstractions and compress the audio space. This is of great interest for sound synthesis for which the latent space has to be kept as low-dimensional as possible (while maintaining a good reconstruction accuracy) in order to be "controlled" by a musician.

Fig. 4 shows that the overall performance of VAEs is in between the performance of DAEs (even equals DAEs for lower encoding dimensions, say smaller than 12) and the performances of PCA and AE. Let us recall that minimizing the reconstruction accuracy is not the only goal of VAE which also aims at constraining the distribution of the latent space. As shown in Fig. 4, the parameter  $\beta$ , which balances regularization and reconstruction accuracy in (2), plays a major role. As expected, high  $\beta$  values foster regularization at the expense of reconstruction accuracy. However, with  $\beta \leq 2 \cdot 10^{-6}$  the VAE clearly outperforms PCA, e.g. up to 20% for  $enc = 12$ .

It can be noticed that when the encoding dimension is high ( $enc = 100$ ), PCA seems to outperform all the other models. Hence, in that case, the simpler (linear model)

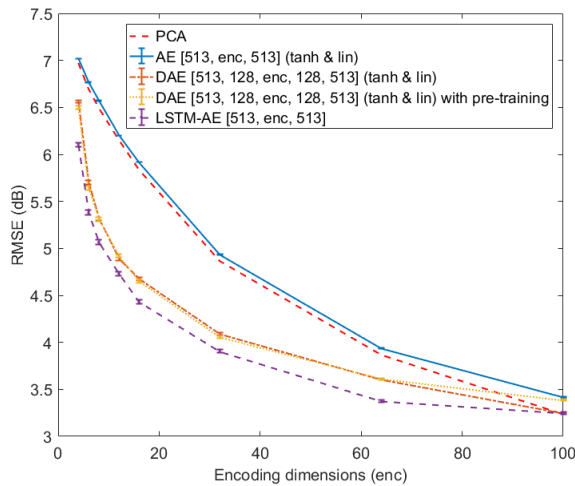


Figure 3: Reconstruction error (RMSE in dB) obtained with PCA, AE, DAE (with and without layer-wise training) and LSTM-AE, as a function of latent space dimension.

seems to be the best (we can conjecture that achieving the same level of performance with autoencoders would require more training data, since the number of free parameters of these model increases drastically). However, using such high-dimensional latent space as control parameters of a music sound generator is impractical.

Similar conclusions can be drawn from Fig. 5 and Fig. 6 in terms of audio quality. Indeed, in a general manner, the PEMO-Q scores are well correlated with RMSE measures in our experiments. PEMO-Q measures for PCA and AE are very close, but PCA still slightly outperforms the shallow AE. The DAEs and the VAEs both outperform the PCA (up to about 11% for  $enc = 12$  and  $enc = 16$ ) with the audio quality provided by the DAEs being a little better than for the VAEs. Surprisingly, and contrary to RMSE scores, the LSTM-AE led to a (slightly) lower PEMO-Q scores, for all considered latent dimensions. Further investigations will be done to assess the relevance of such differences at the perceptual level.

### 3.5 Decorrelation of the Latent Dimensions

Now we report further analyses aiming at investigating how the extracted latent dimensions may be used as *control* parameters by the musician. In the present sound synthesis framework, such control parameters are expected to respect (at least) the following two constraints i) to be as decorrelated as possible in order to limit the redundancy in the spectrum encoding, ii) to have a clear and easy-to-understand perceptual meaning. In the present study, we focus on the first constraint by comparing PCA, DAEs, LSTM-AE and VAEs in terms of correlation of the latent dimensions. More specifically, the absolute values of the correlation coefficient matrices of the latent vector  $\mathbf{z}$  were computed on each sound from the test dataset and Fig. 7 reports the mean values averaged over all the sounds of the test dataset. For the sake of clarity, we present here these results only for a latent space of dimension 16 for one

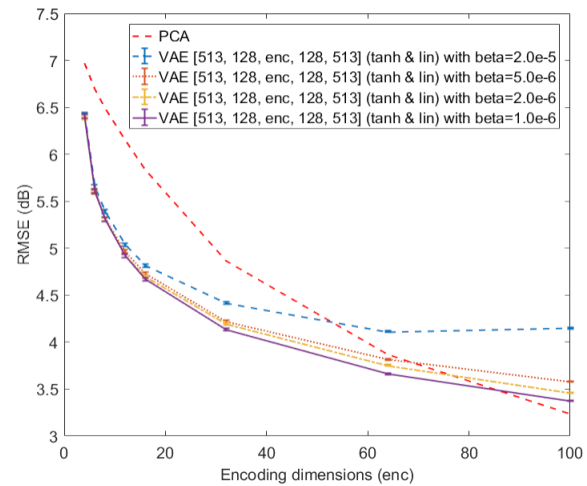


Figure 4: Reconstruction error (RMSE in dB) obtained with VAEs as a function of latent space dimension (RMSE obtained with PCA is also recalled).

model of DAE ([513, 128, 16, 128, 513] (tan & lin) with end-to-end training) and for VAEs with the same architecture ([513, 128, 16, 128, 513] (tan & lin)) and different values of  $\beta$  (from  $1.10^{-6}$  to  $2.10^{-5}$ ).

As could be expected from the complexity of its structure, we can see that the LSTM-AE extracts a latent space where the dimensions are significantly correlated with each other. Such additional correlations may come from the sound dynamics which provide redundancy in the prediction. We can also see that PCA and VAEs present similar behaviors with much less correlation of the latent dimensions, which is an implicit property of these models. Interestingly, and in accordance with (2), we can notice that the higher the  $\beta$ , the more regularized the VAE and hence the more decorrelated the latent dimensions. Importantly, Fig. 7 clearly shows that for a well-chosen  $\beta$  value, the VAE can both extract latent dimensions that are much less correlated than for corresponding DAEs, which makes it a better candidate for extracting good control parameters, while allowing fair to good reconstruction accuracy (see Fig. 4). The  $\beta$  value has thus to be chosen wisely in order to find the optimal trade-off between decorrelation of the latent dimensions and reconstruction accuracy.

### 3.6 Examples of Sound Interpolation

As a first step towards the practical use of the extracted latent space for navigating through the sound space and creating new sounds, we illustrate how it can be used to interpolate between sounds, in the spirit of what was done for instrument hybridization in [9]. We selected a series of pairs of sounds from the NSynth dataset with the two sounds in a pair having different characteristics. For each pair, we proceeded to separate encoding, entry-wise linear interpolation of the two resulting latent vectors, decoding, and finally individual signal reconstruction with inverse STFT and the Griffin and Lim algorithm to reconstruct the phase spectrogram [19]. We experimented dif-

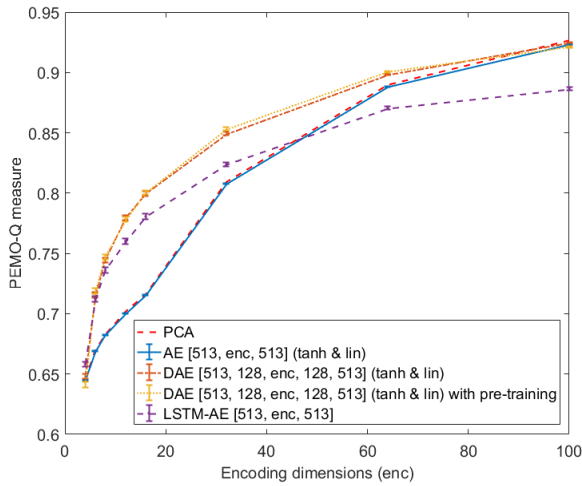


Figure 5: PEMO-Q measures obtained with PCA, AE, DAEs (with and without layer-wise training) and LSTM-AE, as a function of latent space dimension.

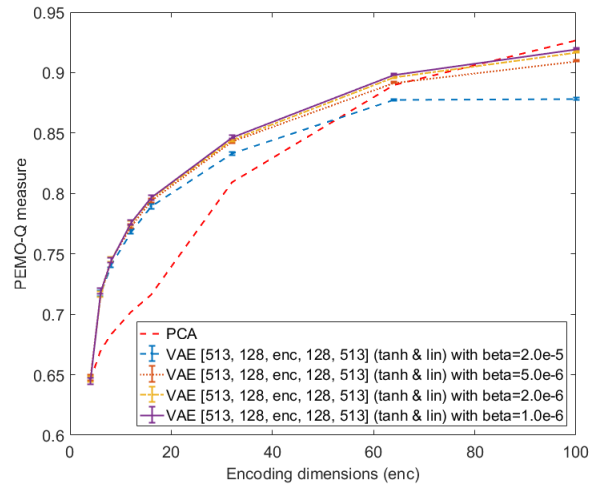


Figure 6: PEMO-Q measures obtained with VAEs as a function of latent space dimension (measures obtained with PCA are also recalled).

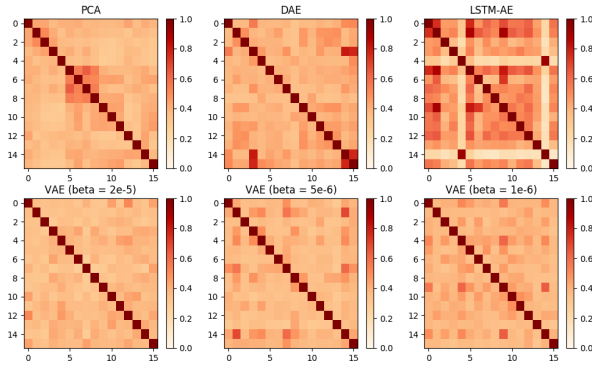


Figure 7: Correlation matrices of the latent dimensions (average absolute correlation coefficients) for PCA, DAE, LSTM-AE and VAEs.

ferent degrees of interpolation between the two sounds:  $\hat{\mathbf{z}} = \alpha \mathbf{z}_1 + (1 - \alpha) \mathbf{z}_2$ , with  $\mathbf{z}_i$  the latent vector of sound  $i$ ,  $\hat{\mathbf{z}}$  the new interpolated latent vector, and  $\alpha \in [0, 0.25, 0.5, 0.75, 1]$  (this interpolation is processed independently on each pair of vectors of the time sequence). The same process was applied using the different AE models we introduced earlier.

Fig. 8 displays one example of results obtained with PCA, with the LSTM-AE and with the VAE (with  $\beta = 1.10^{-6}$ ), with an encoding dimension of 32. Qualitatively, we note that interpolations in the latent space lead to a smooth transition between source and target sound. By increasing sequentially the degree of interpolation, we can clearly go from one sound to another in a consistent manner, and create interesting hybrid sounds. The results obtained using PCA interpolation are (again qualitatively) below the quality of the other models. The example spectrogram obtained with interpolated PCA coefficients is blurrier around the harmonics and some audible artifacts appear. On the opposite, the LSTM-AE seems to outperform the other models

by better preserving the note attacks (see comparison with VAE in Fig. 8). More interpolation examples along with corresponding audio samples can be found at <https://goo.gl/Tvvb9e>.

#### 4. CONCLUSIONS AND PERSPECTIVES

In this study, we investigated dimensionality reduction based on autoencoders to extract latent dimensions from a large music sound dataset. Our goal is to provide a musician with a new way to generate sound textures by exploring a low-dimensional space. From the experiments conducted on a subset of the publicly available database NSynth, we can draw the following conclusions: i) Contrary to the literature on image processing, shallow autoencoders (AEs) do not here outperform principal component analysis (in terms of reconstruction accuracy); ii) The best performance in terms of signal reconstruction is always obtained with deep or recurrent autoencoders (DAEs or LSTM-AE); iii) Variational autoencoders (VAEs) lead to a fair-to-good reconstruction accuracy while constraining the statistical properties of the latent space, ensuring some amount of decorrelation across latent coefficients and limiting their range. These latter properties make the VAEs good candidates for our targeted sound synthesis application.

In line with the last conclusion, future works will mainly focus on VAEs. First, we will investigate recurrent architecture for VAE such as the one proposed in [30]. Such approach may lead to latent dimensions encoding separately the sound texture and its dynamics, which may be of potential interest for the musician.

Then, we will address the crucial question of the perceptual meaning/relevance of the latent dimensions. Indeed using a non-informative prior distribution of  $\mathbf{z}$  such as a standard normal distribution does not ensure that each dimension of  $\mathbf{z}$  represents an interesting perceptual dimension of the sound space, although this is a desirable objec-

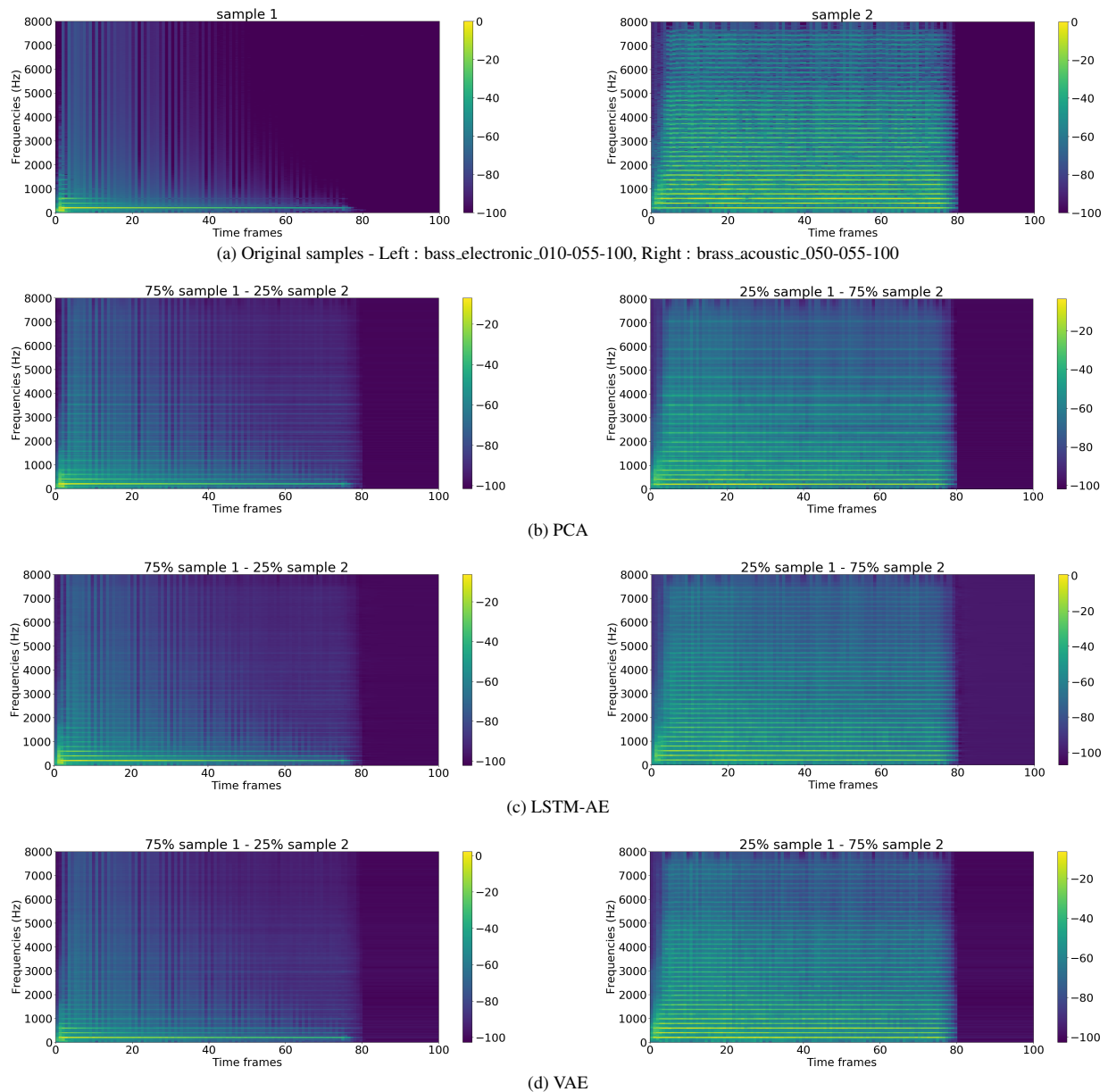


Figure 8: Examples of decoded magnitude spectrograms after sound interpolation of 2 samples (top) in the latent space using respectively PCA (2nd row), LSTM-AE (3rd row) and VAE (bottom). A more detailed version of the figure can be found at <https://goo.gl/Tvvb9e>.

tive. In [14], the authors recently proposed a first solution to this issue in the context of a restricted set of acoustic instruments. They introduced in the variational lower bound (2) of the VAE loss an additional regularization term encouraging the latent space to respect the structure of the instrument timbre. In the same spirit, our future works will investigate different strategies to model the complex relationships between sound textures and their perception, and introduce these models at the VAE latent space level.

## 5. ACKNOWLEDGMENT

The authors would like to thank Simon Leglaive for our fruitful discussions. This work was supported by ANRT in the framework of the PhD program CIFRE 2016/0942.

## 6. REFERENCES

- [1] G. Hinton and R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, Montreal, Canada, 2014.
- [3] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative adversarial text to image synthesis,” in *Proc. of the Int. Conf. on Machine Learning*, New York, NY, 2016.



- [4] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *Int. Conf. on Learning Representations*, 2017.
- [5] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz, "Mocogan: Decomposing motion and content for video generation," in *Proc. of the IEEE conf. on computer vision and pattern recognition*, 2018.
- [6] E. Miranda, *Computer Sound Design: Synthesis Techniques and Programming*, ser. Music Technology series. Focal Press, 2002.
- [7] A. Sarroff and M. Casey, "Musical audio synthesis using autoencoding neural nets," in *Joint Int. Computer Music Conf. and Sound and Music Computing Conf.*, Athens, Greece, 2014.
- [8] J. Colonel, C. Curro, and S. Keene, "Improving neural net auto encoders for music synthesis," in *Audio Engineering Society Convention*, New-York, NY, 2017.
- [9] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi, "Neural audio synthesis of musical notes with wavenet autoencoders," *arXiv preprint arXiv:1704.01279*, 2017.
- [10] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.
- [11] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
- [12] M. Blaauw and J. Bonada, "Modeling and transforming speech using variational autoencoders," in *Conf. of the Int. Speech Comm. Association (Interspeech)*, San Francisco, CA, 2016.
- [13] W.-N. Hsu, Y. Zhang, and J. Glass, "Learning latent representations for speech generation and transformation," *arXiv preprint arXiv:1704.04222*, 2017.
- [14] P. Esling, A. Chemla-Romeu-Santos, and A. Bitton, "Generative timbre spaces with variational audio synthesis," in *Proc. of the Int. Conf. on Digital Audio Effects 2018*, Aveiro, Portugal, 2018.
- [15] S. Leglaive, L. Girin, and R. Horaud, "A variance modeling framework based on variational autoencoders for speech enhancement," in *IEEE Int. Workshop on Machine Learning for Signal Process.*, Aalborg, Denmark, 2018.
- [16] —, "Semi-supervised multichannel speech enhancement with variational autoencoders and non-negative matrix factorization," in *IEEE Int. Conf. on Acoustics, Speech and Signal Process.*, Brighton, UK, 2019.
- [17] L. Li, H. Kameoka, and S. Makino, "Fast MVAE: Joint separation and classification of mixed sources based on multichannel variational autoencoder with auxiliary classifier," in *IEEE Int. Conf. on Acoustics, Speech and Signal Process.*, Brighton, UK, 2019.
- [18] R. Huber and B. Kollmeier, "PEMO-Q: A new method for objective audio quality assessment using a model of auditory perception," *IEEE Transactions on Audio, Speech, and Language Process.*, vol. 14, no. 6, pp. 1902–1911, 2006.
- [19] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Process.*, vol. 32, no. 2, pp. 236–243, 1984.
- [20] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.
- [21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [22] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Process. Systems*, Vancouver, Canada, 2007.
- [23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [24] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.
- [25] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, " $\beta$ -VAE: learning basic visual concepts with a constrained variational framework," in *Int. Conf. on Learning Representations*, 2017.
- [26] F. Chollet *et al.*, "Keras," <https://keras.io>, 2015.
- [27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.
- [28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [29] Y. LeCun, C. Cortes, and C. Burges, "The MNIST database of handwritten digits," <http://yann.lecun.com/exdb/mnist/>, 1998.
- [30] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in Neural Information Process. Systems*, 2015.

# Polytopic reconfiguration: a graph-based scheme for the multiscale transformation of music segments and its perceptual assessment

**Valentin GILLOT**

IRISA / INSA, Rennes (France)  
valentin.gillot@irisa.fr

**Frédéric BIMBOT**

IRISA / CNRS UMR-6074, Rennes (France)  
frederic.bimbot@irisa.fr

## ABSTRACT

Music is a sequential process for which relations between adjacent elements play an important role. Expectation processes based on alternations of similarity and novelty contribute to the structure of the musical flow. In this work, we explore a polytopic representation of music, which accounts for expectation systems developing at several time-scales in parallel. After recalling properties of polytopic representations for describing multi-scale implication processes, we introduce a scheme for recomposing musical sequences by simple transformations of their support polytope. A specific set of permutations (referred to as Primer Preserving Permutations or PPP) are of particular interest, as they preserve systems of analogical implications within musical segments. By means of a perceptual test, we study the impact of PPP-based transformations by applying them to the choruses of pop songs in midi format and comparing the result with Randomly Generated Permutations (RGP). In our test, subjects are asked to rate musical excerpts reconfigured by PPP-based transformations versus RGP-based ones in terms of musical consistency and of attractiveness. Results indicate that PPP-transformed segments score distinctly better than RGP-transformed for the two criteria, suggesting that the preservation of implication systems plays an important role in the subjective acceptability of the transformation. Additionally, from the perspective of building an automatic recomposition system for artistic creation purposes, we introduce, in appendix, the preliminary version of an automatic method for decomposing segments into low-scale musical elements, taking into account possible phase-shifts between the musical surface of the melody and the metrical information.

## 1. INTRODUCTION

Music is usually considered as a sequential process, where sounds, group of sounds and motifs are occurring chronologically, following the natural unfolding of time. Under this approach, music is considered as a flow of information, the organization of which is essentially governed by the relations that develop between adjacent elements. This property of sequentiality is indeed central to many models

of music description and generation, including musicological models such as Schenkerian analysis [1], generative theories based on tree-like structures such as GTTM [2], or computational automata such as Markov chains [3].

From a complementary perspective, it is also commonly accepted that the organization of music is widely based on alternations of similarity and novelty which create patterns and systems of expectation and surprise that ultimately contribute to the structure of the musical discourse from a cognitive point of view [4] [5] [6]. These can be represented as a Polytopic Graph of Latent Relations [7] where each node of the graph represents a low-scale musical segment and vertices correspond to their relation within the expectation systems.

The aim of this work is to explore the polytopic model in terms of its relevance to account for the inner structure of musical segments, by assessing its potential use for the structural recomposition of music. In fact, as presented in section 2, the polytopic representation of a musical segment enables a large range of transformations by applying various permutations to its nodes, thus generating multiple reconfigurations of its musical content, with the same elements in a different order.

Specific permutations, called Primer Preserving Permutations (PPP), are of particular interest, as they preserve systems of analogical implications between metrically homologous elements within the segment [8]. The central hypothesis of the present work is that the musical consistency of PPP-transformed segments will therefore be less affected than it would be by an “ordinary” (i.e. randomly generated) permutation.

In section 2, we describe the implementation of the polytopic reconfiguration process and we elaborate on the organizational properties of Primer Preserving Permutations as well as their potential impact on the inner structure of musical segments.

Then, in section 3, we assess the relevance of the reconfiguration scheme (and its underlying hypotheses): we report on a perceptual test where subjects are asked to rate musical properties of MIDI segments, some of them have been reconfigured with PPPs while others were transformed by Randomly Generated Permutations (RGPs) designed so as to possess a comparable number of discontinuities.

In appendix, we introduce an automatic method for decomposing segments into low-scale musical elements, taking into account possible phase-shifts between the musical surface and the metrical information (for instance, anacrusis).

Copyright: © 2019 Valentin GILLOT et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 2. BACKGROUND AND FORMALISM

### 2.1 Musical objects and time-scales

Before detailing the background and formalism aspects of this work, it is important to specify what musical objects are under consideration in our study. Following Snyder's "levels of musical experience" [9], one may identify three ranges of time-scales in music: an *event-fusion* level, up to  $1/32^{th}$  second (corresponding to early processing), an *intermediate* "melodic and rhythmic grouping" level, between  $1/16^{th}$  and 8 sec. (governed by short-term memory) and a *form* level, 16 sec. or above (resorting to long-term memory).

In this work, we focus on sectional units (i.e. the first time-scale of the form level) and we study their inner organization as the result of the relations existing between successive constituents at the intermediate level, around 1 sec (i.e. typically half-bars). We put a particular focus on melody, with the consequence that our low-scale objects are small groups of a few notes with a common harmonic background.

### 2.2 Polytopal representations of implication systems

The basic concept of a sequential implication system is that the observation of a given event by a subject with no particular preconception will trigger (to that subject) the expectation that the next event is likely to be similar to the first one. This is considered as even more so, in the case of repetition, as is expressed by Narmour's implication principles for the analysis of basic melodic structures [10]:

$$\begin{aligned} A + A &\longrightarrow A \\ A + B &\longrightarrow C \end{aligned}$$

thus meaning that the repetition of two similar patterns  $A$  induces the expectation of a third similar pattern  $A$ , whereas two different patterns  $A$  and  $B$  trigger the expectation of something different from both:  $C$ . Narmour's principles can be understood as a cognitive model based on Gestalt Theory, as mentioned in [11].

This sequential implication principle can be extended by now considering a *system* of elements viewed as forming the base of an analogical induction process [12]. For instance, consider the matrices below, and imagine what could be the missing elements in terms of "logical" implications:

A	B	2	4	$\cap$	$\downarrow$
E	.	8	.	$\cup$	.

As developed in [6], such square systems trigger an expectation process which spans over 2 time-scales simultaneously and which writes, in a formalism compatible with Narmour's:

$$A + f(A) + g(A) \longrightarrow g(f(A))$$

and in particular:

$$\begin{aligned} A + B + A &\longrightarrow B \\ A + A + B &\longrightarrow B \end{aligned}$$

These two prototypical cases of square implication systems are indeed the basis of two very frequent structural patterns,  $ABAC$  and  $AABC$ , which can be understood as the denial, in  $4^{th}$  position, of the implication system triggered by the first 3 elements. But quite a number of other such redundant patterns can be embedded in this framework.

Further generalizing this principle by encompassing more time-scales, as in [13], leads to cubic (3-scale), tesseract (4-scale) and more generally  $n$ -cubic systems where  $n$  time-scales are considered simultaneously (see Figure 1).

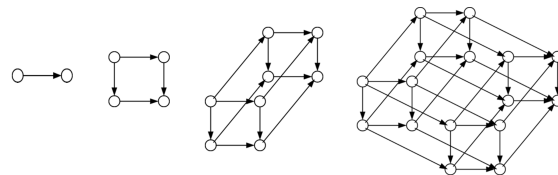


Figure 1. Graph representation of multi-scale implication systems. From left to right: segment, square, cube, tesseract.

When mapped with a sequence of musical events, these polytopal graphs can be used to represent analogical relations between musical objects in metrically homologous positions at different time-scales [7].

### 2.3 Structural reconfiguration by node permutations

There are many ways to define a permutation, but the most adequate one in the context of this work is to view it as a bijective function  $\varphi$  (i.e. a one-to-one correspondence) between the  $N$  time indexes of an original sequence  $x_0 \dots x_{N-1}$  and those of the permuted sequence  $x_{\varphi(0)} \dots x_{\varphi(N-1)}$ .

Applying a permutation to the time indexes creates a certain degree of disruption in the original flow of the sequence. In this paper, we characterize this fact by two properties:  $D$ , the number of *discontinuities* created by the permutation and  $E$ , the ( $\log_2$  of the) maximum time-interval (or *excursion*) between 2 consecutive elements in the permuted sequence.

$$\begin{aligned} D &= \#\{t \mid [\varphi(t) - \varphi(t-1)] \neq 1\}_{0 \leq t < N} \\ E &= \log_2 \max_{0 \leq t < N} |\varphi(t) - \varphi(t-1)| \end{aligned}$$

Among all possible permutations  $\varphi$ , a particular subset of them is focused on in this work. They are referred to as PPPs (for *Primer Preserving Permutations*) and were introduced by Louboutin et al. [8] as the set of permutations which preserve the systemic relations of the musical elements  $x_t$  in the sequence, by just interchanging the time-scales at which they develop. These permutations are therefore particularly interesting to investigate on the relevance of the implications systems (and their preservation) in the consistency of a musical segment.

In the case of a sequence of  $N=16$  elements, the polytope is a 4-cube, also called a *tesseract* (see Figure 2), and it can be viewed as being composed of 4 homologous lower-scale systems of 4 elements (formed by 4 parallel faces of the tesseract), themselves linked by an upper-scale system

PPP																		Disc.	Exc.
ORIG	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		0	1
1	0	2	4	6	1	3	5	7	8	10	12	14	9	11	13	15	14	1	1
2	0	1	8	9	2	3	10	11	4	5	12	13	6	7	14	15	7	2	2
3	0	1	4	5	2	3	6	7	8	9	12	13	10	11	14	15	6	1	1
4	0	4	8	12	1	5	9	13	2	6	10	14	3	7	11	15	15	3	3
5	0	2	8	10	1	3	9	11	4	6	12	14	5	7	13	15	15	2	2

Table 1. Extensive definition of the 6 PPPs and their properties (discontinuity and excursion - see text).

RGP																		Disc.	Exc.
1	0	3	4	2	7	1	6	5	8	13	10	9	11	14	12	15		14	1
2	0	4	5	6	1	3	8	9	10	11	12	13	14	2	7	15		7	2
3	0	1	2	5	3	4	6	7	8	9	10	12	13	11	14	15		6	1
4	0	5	3	8	6	4	14	2	10	1	9	12	7	11	13	15		15	3
5	0	9	7	1	6	4	2	8	10	3	11	14	13	5	12	15		15	2

Table 2. Examples of RGPs and their properties (discontinuity and excursion - see text).

formed by the first element of each lower-scale system (on a face which is perpendicular to the 4 lower-scale ones) (see Figure 3).

Mapping these systems of faces by a permutation, while preserving time order within faces, yields 5 possibilities (plus the identity), which correspond to the PPPs defined in Table 1. Note that each PPP maps 0 to 0 and  $N-1$  to  $N-1$ .

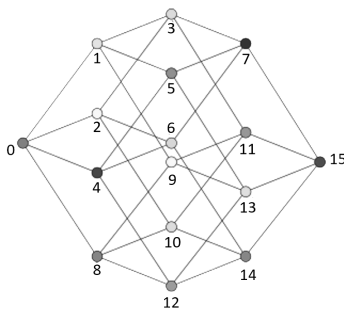


Figure 2. Example of the polytopal representation of a time sequence in the case of a tesseract.

By construction, PPPs preserve implication systems of 4 elements in the sense that sets of 4 parallel faces in the tesseract are mapped on another set of parallel faces. By doing so, analogical systems are preserved, and only the scales at which they develop are modified: for instance,  $[x_0x_1x_2x_3]$  in the original is “transferred” to  $[x_0x_4x_1x_5]$  with PPP1 and  $[x_0x_4x_8x_{12}]$  with PPP4. Moreover, the last element of a system never occurs before all the elements of all the systems it belongs to, have themselves occurred, therefore preserving a principle of causality in all the analogical implications.

For comparison purposes, Table 2 illustrates 5 *Randomly Generated Permutations* (RGPs), which also start with 0, end with  $N-1$ , and have the same profile as their corresponding PPPs in terms of discontinuity and excursion. It can be easily checked that, unless incidentally, analogical systems are not preserved by RGPs.

Ultimately, a given permutation is applied to a musical se-

quence of events by reading and copying (in the permuted order) the original musical material after having segmented into proper low-scale elements. Figure 4 illustrates this process on a 8-bar melodic line (original at the top), processed as a sequence of  $N=16$  half-bar (2-beat) elements, resulting in reconfigured melodic lines, the organization of which is either driven by PPPs or by RGPs.

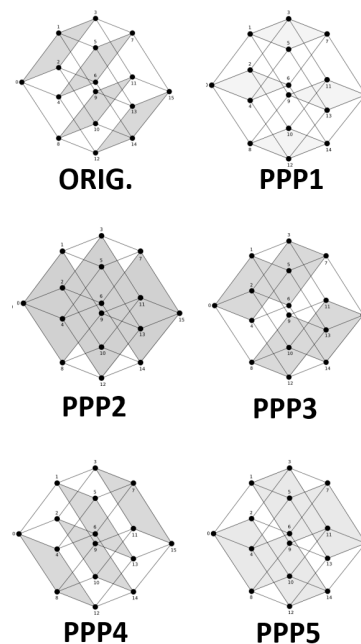


Figure 3. Representation of the 6 systems of parallel faces on the tesseract and their corresponding PPP in reference to the original.

## 2.4 Handling musical surface time-shifts

As a prerequisite to the application of a permutation to a sequence of musical objects, it is of primary importance to define precisely the location and boundaries of the objects that will be subjected to the transformation.

In this work, elementary objects are melodic fragments



Original

PPP 1

PPP 2

PPP 3

PPP 4

PPP 5

RGP 1

RGP 2

RGP 3

RGP 4

RGP 5

Figure 4. Example of the 11 types of reorganized variants on a 8-bar melodic line.

of 1/2-bar long which correspond to 2 beats in the case of a 4-4 meter (as is the case in the example of Fig 4).

However, for reconfiguration purposes, these “objects” may not be optimal if they are blindly synchronized on downbeats (and sub-downbeats). In a number of cases, the musical surface of the “proper” low-scale melodic elements should be slightly shifted in order to correspond to an optimal grouping of the notes and therefore a better result after permutation.

This situation is illustrated in Figure 5 where the first melodic element should be considered as starting 1.5 beat before the first downbeat of the section.

In designing the perceptual test described in the next section, this optimal time-shift has been adjusted manually on a case-by-case basis. However, we present in the appendix, a preliminary algorithm that has been developed to estimate the time-shift automatically.

### 3. PERCEPTUAL EXPERIMENTS IN STRUCTURAL RECONFIGURATION

In this section, we present the details of the methodology which has been used for the evaluation of polytopic reconfigurations. The general principle has been to design a perceptual test where subjects are presented with musical excerpts which are either PPP-transformed or RGP-transformed, and where they are asked to rate (blindly) these excerpts both in terms of consistency and attractive-

ness. We then study the difference of ratings across subjects between the two types of transformations.

#### 3.1 Test data

The experiments are performed on the RWC POP corpus, a subset of the Real World Computing (RWC) music database created for scientific purposes by Goto et al. [14]. This dataset has been designed for researchers. It is available for a small fee and without restrictions.

The RWC POP set is made of 100 songs in WAV and MIDI formats. According to the authors, they were composed partly in the style of the 80s American hits and partly in the 90s Japanese style. This corpus is very commonly used in various task in Music Processing and Information Retrieval.

Using the RWC songs in our tests has an additional advantage: there is virtually no risk that subjects participating to the test already know the songs from which the excerpts are stemming. This is a favorable situation as the familiarity with a particular song could affect the rating of the transformed excerpts in an uncontrolled way.

For our experiments, a subset of 24 songs has been selected for which the first instance of chorus is exactly 8-bar long. The reconfiguration process is applied to the MIDI version of the chorus. The elements of interest are the melodic lines, the accompaniment (harmony and bass) and the drums. We consider that these elements provide an acceptable rendering of the structural information of the



Figure 5. Example of a melodic line starting 1.5 beat before the first downbeat of the section.

chorus and they all are transformed synchronously by the permutations. The average duration of a chorus is 19.5 seconds.

As presented in section 2, these transformations are performed on the original chorus after having segmented it into 16 elements (with a possible shift of the musical surface of the melody, as explained in section 2.4). Then, the 10 permutations (5 PPPs and 5 RGPs) are applied to the 16 elements, resulting in a re-ordering of the original choruses, i.e. 10 new musical contents (per chorus), which we will call “variants” of the original.

To summarize, for each of the 24 choruses, the following 11 variants are considered:

- the original version of the chorus
- 5 PPP-transformed variants (PPP<sub>1</sub> to PPP<sub>5</sub>)
- 5 RGP-transformed variants (RGP<sub>1</sub> to RGP<sub>5</sub>)

The perceptual test is therefore conducted on  $24 \times 11 = 264$  distinct excerpts.

### 3.2 Protocol

The perceptual experiments are performed with 66 subjects, selected among colleagues and relatives of the authors<sup>1</sup>. About 25% of the subjects are female and 75% male, aged between 22 and 55 years old.

In order to avoid the bias that could arise if a subject gets familiar with a particular chorus, each subject is presented with one and only one variant of each of the 24 choruses, in a random order. The total number of tests in our experiments is therefore equal to  $66 \times 24 = 1584$ , i.e. 144 tests per variant. Moreover, the 11 variants are distributed as evenly as possible for each subject, i.e. typically 2 (or occasionally 3) per subject.

After a brief general presentation of the test (during which the subjects are *not* provided with any detail on the scientific motivation of the test), they are asked to answer, for each excerpt, two questions formulated as follows:

1. “Please give your opinion as of the consistency of the musical construction of this passage”.
2. “Please give your opinion as of the degree of musical attractiveness of this passage”

The subjects must respond by placing a cursor on a 5-level scale corresponding to 5 different levels of evaluation (from *very bad consistency* to *very good consistency* and from *very unattractive* to *very attractive*). They also have the possibility to position the cursor between two graduations, thus offering them 9 levels of rating altogether.

<sup>1</sup> For obvious reasons, the authors themselves were not included in the panel of subjects.

Asking two different questions aims at leading subjects to decorrelate as much as possible their rating of the acceptability of the musical construction (question 1), from their own personal taste (which they can express in their response to question 2).

### 3.3 Scoring

For each subject, we thus collect 24 consistency ratings and 24 attractiveness ratings.

Let us denote as  $r$ , the generic variable which designates such ratings (ignoring for the moment whether it is *consistency* or *attractiveness*, for the sake of notations’ simplicity).

The 24 ratings  $r_{jk}$  produced by subject  $j$  ( $1 \leq j \leq 66$ ,  $1 \leq k \leq 24$ ) are distributed into the 11 variants:  $\mathcal{O}$  for original,  $\mathcal{P}_i$  for PPP <sub>$i$</sub>  ( $1 \leq i \leq 5$ ) and  $\mathcal{P}_i^*$  for RGP <sub>$i$</sub>  ( $1 \leq i \leq 5$ ). These individual ratings are averaged separately, thus leading to 11 mean ratings (one per type of variant):

$$\mu_{0j} = \frac{1}{m_{0j}} \sum_{\mathcal{O}} r_{jk} \quad \text{for the original choruses} \quad (1)$$

$$\mu_{ij} = \frac{1}{m_{ij}} \sum_{\mathcal{P}_i} r_{jk} \quad \text{for the 5 PPP variants} \quad (2)$$

$$\mu_{ij}^* = \frac{1}{m_{ij}^*} \sum_{\mathcal{P}_i^*} r_{jk} \quad \text{for the 5 RGP variants} \quad (3)$$

Note that, in practice,  $m_{0j}$ ,  $m_{ij}$  and  $m_{ij}^*$  are equal to 2 or 3 and their sum is equal to 24.

To evaluate the relative degradation created by the various reconfigurations of the music material, we can measure:

$$\delta_{ij} = \mu_{0j} - \mu_{ij} \quad (4)$$

$$\delta_{ij}^* = \mu_{0j} - \mu_{ij}^* \quad (5)$$

namely the signed difference, for a given subject  $j$ , between his/her ratings of the reconfigured variants and his/her ratings of the originals.

For each subject and variant, we can also compute:

$$\pi_{ij} = \mu_{ij} - \mu_{ij}^* = \delta_{ij} - \delta_{ij}^* \quad (6)$$

which measures the (positive or negative) preference of PPP <sub>$i$</sub> -transformed excerpts over RGP <sub>$i$</sub> -transformed ones, for speaker  $j$  (where PPP <sub>$i$</sub>  and RGP <sub>$i$</sub>  are comparable in terms of discontinuities  $D$  and excursion  $E$ ).

Ultimately, we focus on 3 scores:

$$\pi_j = \frac{1}{5} \sum_{i=1}^5 \pi_{ij} \quad (7)$$

$$\delta_i = \frac{1}{66} \sum_{j=1}^{66} \delta_{ij} \quad (8)$$

$$\delta_i^* = \frac{1}{66} \sum_{j=1}^{66} \delta_{ij}^* \quad (9)$$

where  $\pi_j$  provides a global preference score (positive or negative) on PPPs over RGP for each subject  $j$ , while  $\delta_i$  (resp.  $\delta_i^*$ ) provides a degradation score for each of the PPP- (resp. RGP-) reconfigured variants, averaged over all subjects.

These two quantities are calculated separately for consistency ratings and for attractiveness ratings.

### 3.4 Results

Figure 6 and 7 depict the distributions of consistency and attractiveness preference scores ( $\pi_j$ ), ranked from lowest to highest, over the panel of 66 subjects. In both cases, the distributions are visibly shifted towards positive values, especially for consistency ratings. Indeed, for consistency, 57.5 subjects have a positive preference for PPPs versus 8.5 having a negative preference<sup>2</sup> (i.e. 87.1 % vs 12.9 %). The average score for positive judgments amounts to +0.89 whereas it rests at -0.30 only, for negative judgments. For attractiveness, the proportions are still clearly in favor of PPPs, but not as contrasted (74.2 % vs 25.8 %), with average scores of +0.74 and -0.55 respectively.

Figure 8 and 9 represent the relative degradation scores ( $\delta_i$  and  $\delta_i^*$ ) for the 10 variants, ranked in increasing order of degradation. From the ranking of the permutations, it is noticeable that the degradation score is primarily correlated with the number of discontinuities of the permutations, as the two permutations with lower  $D$  show a smaller value of  $\delta$  as opposed to the three others.

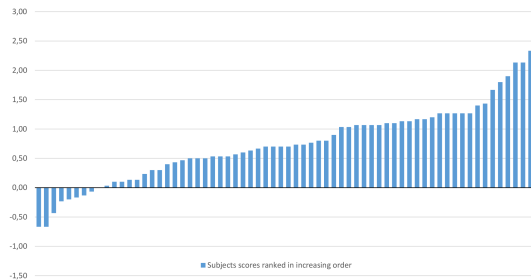


Figure 6. Musical consistency: relative PPP-preference  $\pi_j$  of the test subjects, ranked in increasing order

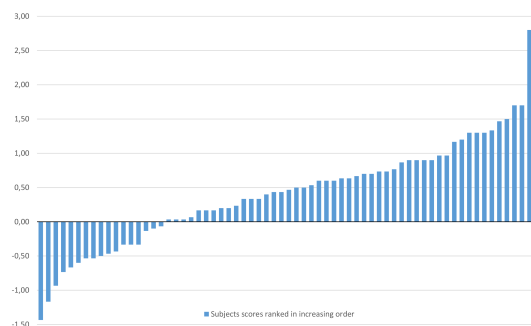


Figure 7. Musical attractiveness: relative PPP-preference  $\pi_j$  of the test subjects, ranked in increasing order

Globally, these results indicate that PPP-based reconfigurations tend to be less disruptive than RGPs against the

perceived consistency of musical segments, thus supporting the hypothesis that the preservation of expectation systems contributes to their inner structure.

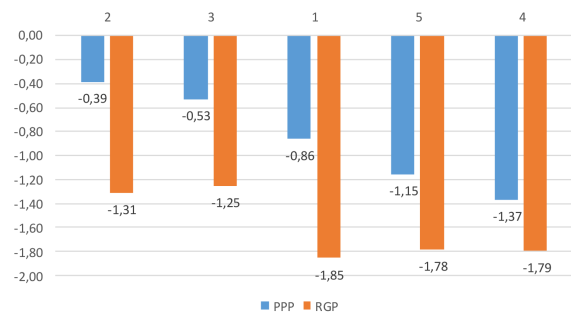


Figure 8. Musical consistency: degradation scores  $\delta_i$  and  $\delta_i^*$  for the different permutation variants (PPP and RGP)

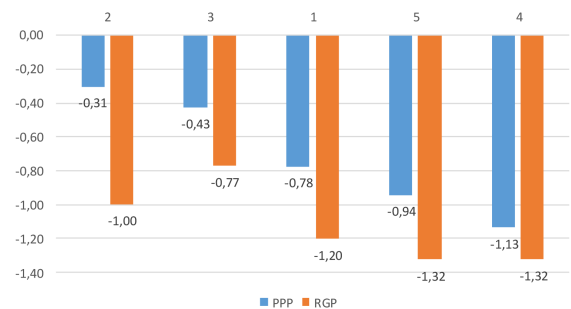


Figure 9. Musical attractiveness: degradation scores  $\delta_i$  and  $\delta_i^*$  for the different permutation variants (PPP and RGP)

## 4. CONCLUSIONS

Systems of relations between elements play a key role in structural constructions in many cognitive dimensions. The experiments reported in this paper constitute an initial investigation on the role of analogical systems (modeled by polytopic graphs) in the perception of segmental structure in pop music.

These results indicate a very noticeable trend towards the hypothesis that the preservation of the analogical implication systems in musical segments impacts positively the perception of their inner structure and hence, their acceptability. However, more extensive tests on a larger population and with more varied musical excerpts are certainly needed to consolidate these first results, and refine their precise scope.

In parallel, we intend to develop the proposed reconfiguration scheme into a music creation tool which could be used by composers to experiment new possibilities in employing musical material with multiple purposes, results and effects. In this context, the polytopic representation is an additional advantage, as it is bound to make manipulation interfaces more intuitive and easy to use.

Ultimately, the development of an automatic segmentation process is a complementary goal towards a fully operational concept.

<sup>2</sup> counting as 0.5 in each category, the subject who scores exactly 0.

## Appendix

### Melodic segmentation with a compressibility criterion

In order to perform a relevant permutation-based transformation to a sequence of musical objects, it is crucial to define precisely the location and boundaries of the objects on which the transformation is applied.

As mentioned in section 2.4, it is quite common that the structure of the melodic line is time-shifted with respect to the phase of the metrical pulsation. As a consequence, it is important to take into account this time-shift when defining the low-level melodic objects. In this section, we present a preliminary method to automatically estimate this time-shift.

For doing so, we estimate a segmentation of the melodic surface by optimizing the grouping of the notes using a compressibility criterion.

### Meter, segmentation and phase-shift

For simplicity and consistency with the rest of our work, we consider musical segments which are 8-bar long, and 4/4 meter, i.e  $L = 32$  beats, and we assume that we want to segment the excerpt in  $m$  elementary objects. In this work  $m = 16$ .

Let  $S$  be the sequence of instants where the strong beats fall, according to the meter and the bar:

$$S = (t_0, t_1, \dots, t_p, \dots, t_{m-1}) \quad (10)$$

Let us now denote as  $\Sigma$  another segmentation that is not necessarily synchronized with downbeats.

$$\Sigma = (\tau_0, \tau_1, \dots, \tau_p, \dots, \tau_{m-1}) \quad (11)$$

In the general case we can write:

$$\tau_p = t_p + \theta_p \quad (12)$$

where  $\theta_p$  is the phase-shift of the musical surface of segment  $p$  in relation to the strong beat location  $t_p$ .

In compliance with the work reported in the body of this article, we further assume that  $\theta_p$  does not depend on  $p$ :

$$\tau_p = t_p + \theta \quad (13)$$

### Properties of the phase-shift

When  $\theta = 0$ , the melodic segments are considered to be starting synchronously with the downbeat. If  $\theta > 0$ , this means that the melodic objects display some phase-delay with the beat, whereas if  $\theta < 0$ , they start in anticipation (as in the case of an anacrusis).

Let  $\alpha$  be the “size” of one elementary object in terms of the number of beats, i.e.:

$$\alpha = \frac{L}{m} \quad (14)$$

Here  $\alpha = 2$ .

Quite naturally, we assume that the phase-shift cannot exceed the size of the elementary objects, i.e.:

$$-\alpha \leq \theta \leq \alpha \quad (15)$$

Fig. 10, shows the segmentation of a 8-bar melody, represented in MIDI format. The first 3 notes of this section are located with a phase-shift of -1,5 (beat) with respect to the metric boundaries: the thick lines correspond to the downbeat instants and the thin ones correspond to the segmentation obtained with  $\theta = -1.5$  (beat).

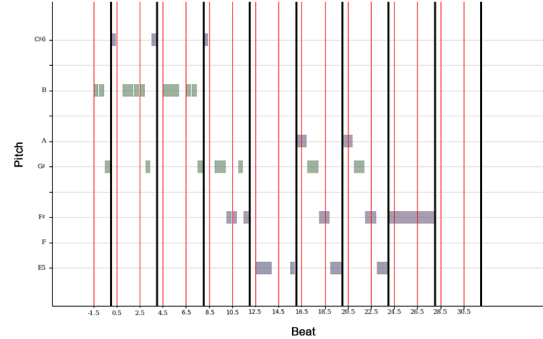


Figure 10. Example of a melodic line segmentation starting 1.5 beat before the first downbeat of the section ( $\theta = -1.5$  beat).

### A compressibility criterion for segmentation

In order to estimate an optimal value of  $\theta$ , we define a compressibility criterion which scores how redundant is the sequence of elements resulting from a given segmentation.

This approach is inspired from Kolmogorov’s information theory [15] and relates, in its spirit, to recent work exploring the potential of approaches based on complexity and compression for modeling music contents (for instance [16] [17]).

Let now  $\Sigma$  be a segmentation as in equation 11 and  $\sigma_p$  the elementary object of  $\Sigma$  within the time interval  $[\tau_p, \tau_{p+1}]$ .

We first define the *parallel run-length* between two melodic elements as the relative duration  $d \in [0, 1]$  during which they remain similar to each other (i.e. identical, up to a translation). We denote as  $b$  the translation between the two similar fragments and we assign to  $b$  a binary value of 0 (if no translation) or 1 (when a translation is observed).

We then define an elementary cross-compressibility score function  $c(\sigma_p, \sigma_q)$  between two melodic objects  $\sigma_p$  and  $\sigma_q$ , by combining  $d$  and  $b$ :

$$c(\sigma_p, \sigma_q) = d(\sigma_p, \sigma_q) - \lambda b(\sigma_p, \sigma_q) \quad (16)$$

For the time being,  $\lambda$  is tuned empirically.

We then define a compressibility score function for each elementary object  $\sigma_p$  within  $\Sigma$ , which depends on the previous objects in the segmentation, for instance:

$$z(\sigma_p) = \frac{1}{p} \sum_{h < p} c(\sigma_h, \sigma_p) \quad (17)$$

Finally, we compute the entire segmentation compressibility score  $Z(\Sigma)$  as:

$$Z(\Sigma) = \frac{1}{m} \sum_p z(\sigma_p) \quad (18)$$



Because  $\theta$  is the same across segments, all possible segmentations  $\Sigma$  can be tested exhaustively, and the optimal segmentation is chosen as the one with the highest compressibility score.

Fig. 11 depicts the behavior of  $Z$  for a particular example, with  $\theta$  ranging from  $-2$  to  $2$  by steps of  $0.25$  beat. Score values are normalized to 1 for the maximum value.

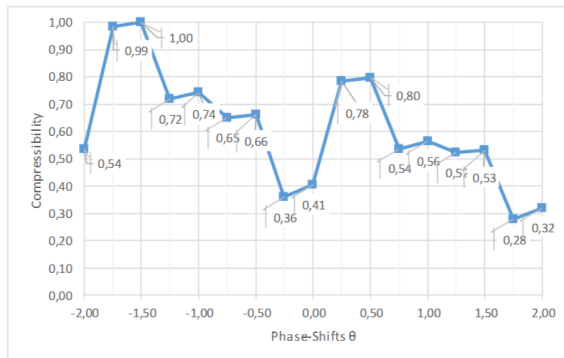


Figure 11. Behavior of the compressibility score  $Z$  for a particular 8-bar melody, with  $\theta$  ranging from  $-2$  to  $2$  beats.

### Current performance level

In its current version, the results provided by this algorithm have been evaluated over the 24 choruses used in the perceptual experiments, by comparing the automatic estimations of the melodic phase-shifts with those determined by a manual annotation: in 10 cases (out of 24), the automatic algorithm provided the same result as the expert.

This approach is currently being improved but the current results encourage us to further investigate the method, so as to evaluate its full potential.

## 5. REFERENCES

- [1] H. Schenker, "Der freie Satz," Vienna: Universal Edition. (translated 1979), 1935.
- [2] F. Lerdahl and R. S. Jackendoff, *A generative theory of tonal music*. MIT press, 1985.
- [3] M. T. Pearce, *The construction and evaluation of statistical models of melodic structure in music perception and composition*. City University London, 2005.
- [4] E. Narmour, "Music expectation by cognitive rule-mapping," *Music Perception: An Interdisciplinary Journal*, vol. 17, no. 3, pp. 329–398, 2000.
- [5] D. B. Huron, *Sweet anticipation: Music and the psychology of expectation*. MIT press, 2006.
- [6] F. Bimbot, E. Deruty, G. Sargent, and E. Vincent, "System & Contrast : A Polymorphous Model of the Inner Organization of Structural Segments within Music Pieces," *Music Perception*, vol. 33, pp. 631–661, June 2016, former version published in 2012 as Research Report IRISA PI-1999, hal-01188244.
- [7] C. Louboutin and F. Bimbot, "Polytopic Graph of Latent Relations: A Multiscale Structure Model for Music Segments," in *6th International Conference on Mathematics and Computation in Music (MCM 2017)*, ser. Lecture Notes in Computer Science book series, O. A. Agustín-Aquino, E. Lluís-Puebla, and M. Montiel, Eds., vol. 10527. Mexico City, Mexico: Springer, Jun. 2017. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-01653445>
- [8] —, "Description of Chord Progressions by Minimal Transport Graphs Using the System & Contrast Model," in *ICMC 2016 - 42nd International Computer Music Conference*, Utrecht, Netherlands, Sep. 2016. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-01421023>
- [9] B. Snyder and R. Snyder, *Music and memory: An introduction*. MIT press, 2000.
- [10] E. Narmour, *The analysis and cognition of melodic complexity: The implication-realization model*. University of Chicago Press, 1992.
- [11] E. Morgan, A. Fogel, A. Nair, and A. D. Patel, "Statistical learning and gestalt-like principles predict melodic expectations," *Cognition*, vol. 189, pp. 23 – 34, 2019. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0010027718303317>
- [12] F. Bimbot, E. Deruty, G. Sargent, and E. Vincent, "Semiotic structure labeling of music pieces: concepts, methods and annotation conventions," in *Proc. ISMIR*, 2012.
- [13] E. Deruty, F. Bimbot, and B. Van Wymeersch, "Methodological and musicological investigation of the System & Contrast model for musical form description," INRIA, Research Report RR-8510, 2013, hal-00965914.
- [14] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, Classical and Jazz Music Databases," in *ISMIR*, vol. 2, 2002, pp. 287–288.
- [15] A. N. Kolmogorov, "Three approaches to the quantitative definition of information," *Problems of information transmission*, vol. 1, no. 1, pp. 1–7, 1965.
- [16] D. Meredith, "Music analysis and Kolmogorov complexity," in *Proceedings of the 19th Colloquio di Informatica Musicale (XIX CIM)*, Trieste, Italy, 2012.
- [17] C. Guichaoua, "Modèles de compression et critères de complexité pour la description et l'inférence de structure musicale," Ph.D. dissertation, Université Rennes 1, 2017.

# Non-Linear Contact Sound Synthesis for Real-Time Audio-Visual Applications using Modal Textures

**Martin Maunsbach**  
Aalborg University  
mmauns17@student.aau.dk

**Stefania Serafin**  
Aalborg University  
sts@create.aau.dk

## ABSTRACT

Sound design is an integral part of making a virtual environment come to life. Spatialization is important to the perceptual localization of sounds, while the quality determines how well virtual objects come to life. The implementation of pre-recorded audio for physical interactions in virtual environments often requires a vast library of audio files to distinguish each interaction from the other.

This paper explains the implementation of a modal synthesis toolkit for the Unity game engine to automatically add impact and rolling sounds to interacting objects. Position-dependent sounds are achieved using a custom shader that can contain textures with modal weighting parameters.

The two types of contact sounds are synthesized using a mechanical oscillator describing a mass-spring system. We describe the discretization methods adopted, the solution of the nonlinear interaction and an implementation in the Unity game engine.

## 1. INTRODUCTION

High quality audio effects for virtual environments, as seen in video games, are important to the user's feeling of presence and overall experience. Specifically, impact sounds of colliding objects are of great importance to games [1], but sound of friction and rolling are also in demand. Sound effects are, however, slow and difficult to create and require specialized talents to implement correctly [2]. Furthermore, lacking realism in the sensory experience of any virtual environment leads to a break in perceived presence. For example, if a table is struck with a hammer and only produces a slight tapping sound, the experience is sending conflicting information to its user. To counter this problem, the field of physical modelling of sound effects is of particular interest.

Through this field of knowledge, the automatic generation of contact sounds can be synthesized without the need for large libraries of pre-recorded samples.

One popular synthesis technique adopted to simulate several material properties is modal synthesis [3,4]. In modal synthesis, the frequencies of vibration in a material are

simulated considering the normal modes of real objects. The normal modes describe the peaks in the spectral contents of a sound. This form of physical modelling of sound is also easily scaled in its level of detail, since the number of frequencies that are modelled at any time can be altered as necessary, and is computationally efficient. This makes it particularly well-suited for real-time implementation [2]. The modes are not only different because of material, but also the shape of the object in question and the position of impact. To accommodate for the position-dependency, a weighting ratio of each mode changes depending on the position of impact. The physical properties of simulated materials are of great importance. For example, materials that are more stiff will produce inherently higher pitched sounds when struck.

In this paper, the design and implementation of a modal synthesis toolbox for the game engine Unity is described. The simulation is produced to free developers using the game engine from the time-consuming process of capturing libraries of impact and rolling sound events, and instead have these sounds rendered through a mechanical oscillator, based on events within their virtual environments.

## 2. RELATED WORK

The fields of computer graphics and physical modelling of sound synthesis often overlap. The finite element method, though computationally heavy, can calculate the modes of 3D models, as seen in a program like mesh2faust, than can compute the modes from a mesh [5]. "Example-guided" automation to modal synthesis has been done, where recordings estimate the physical parameters across an example object. For other objects using the same material - but different geometric shape, the parameters of the example object can be used to automatically transfer the modes for object of differently shaped object [6]. A method modelling the wave propagation of a mesh is the digital waveguide mesh, that uses bidirectional delay units to simulate the reflections and transmissions through connected digital waveguides at a wave impedance [7].

Impact sounds have been synthesized in various ways. One example uses banded digital waveguide synthesis, where dynamically filtered white noise is passed through bandpass filters to add the characteristics of a material [1]. The mechanical oscillator used in this paper is not only useful for impact sounds, but can also other interactions like friction for a rolling wheel, finger rubbing on glass and squeaking doors [8]. The rolling and rubbing interac-

Copyright: © 2019 Martin Maunsbach et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

tion was further explored by Conan et al., where the same equation for the force that explains the sphere-plane interaction in this project also was used [9]. By utilizing the sparsity of modal sounds in the frequency domain, some modal synthesis was found to be 5-8 times faster compared to the time-domain equivalent [10].

When combined with computer graphics, sound synthesis has used shaders to design physical models of massive digital instruments [11], where the GPU is used to help with the computational load. Sound signals have also been stored as RGBA sound signals to use GPU processing [12]. Similar work storing model parameters in textures was done in 2001 by scanning objects using a highly automated robotic facility [13].

Both proprietary and publicly available game engines have begun incorporating synthesis into their sound design. Rockstar's RAGE engine, that was used to develop Grand Theft Auto V, has a real-time synthesis toolkit for assets "you can't necessarily create with samples alone" [14]. The motivation for real-time synthesis was based on dynamic assets, fidelity and memory usage. The publicly available game engine Unreal Engine has also begun adding real-time synthesis to their game engine, and is looking into computing it on the GPU in a way that can be referred to as audio "shaders" [15].

### 3. SYSTEM MODELLING

A mechanical oscillator can take the form [16]:

$$\ddot{x} + g\dot{x} + \omega^2 x = \frac{1}{m}f \quad (1)$$

where the oscillator displacement  $x$  is used to produce the audio output. The frequency is set in  $\omega$  and the damping constant  $g$  is determined by a quality factor  $q$  by  $g = \omega/q$ . The oscillator velocity and acceleration are determined by  $\dot{x}$  and  $\ddot{x}$  respectively. The modal weighting is determined by  $1/m$ . Using the K method to eliminate the delay-free [17] loop that will arise from the force equation and the trapezoidal rule, the output can be discretized take the form

$$w[n] = H(Cy[n] + y[n-1]) + H(\alpha I + A)w[n-1] \quad (2)$$

where  $w$  is the vector  $[x; \dot{x}]$ ,  $y$  is the force  $f$  and  $A$ ,  $C$  and  $H$  are transformation matrices found from Equation (1)

$$H = \frac{1}{\alpha^2 + \alpha g + \omega^2} \cdot \begin{bmatrix} \alpha + g & 1 \\ -\omega^2 & \alpha \end{bmatrix}; \quad (3)$$

$$A = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -g \end{bmatrix}; \quad C = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}$$

If the frequency, quality factor and modal weight remains constant throughout the interaction, these matrices need only be computed once.

The contact force is given by [18]

$$f(x, \dot{x}) = x^\alpha (k + \lambda \dot{x}) \quad (4)$$

where  $x$  is the same oscillator displacement as in Equation (1). This mutual dependency creates the delay-free

loop, where the K method can be used. The velocity at the moment of impact is  $\dot{x}$ , while the constant  $k$  is the elastic coefficient and  $\alpha$  is a value between 1.5 and 3.5 that describes the surface geometry [16]. A value of 1.5 is used in this paper, which leads to the non-linearity, that is solved by approximating the value using Newton-Rhapson.

The modal weight at specific positions can be changed with the weighting factor  $1/m$  in Equation (1). The position-dependency means the matrices that previously only had to be computed once on impact have to be computed every time the modal weight changes. This adds additional constant multiplication for each buffer window, that is negligible in the overall computation cost, as the modal weight only appears in the  $C$  transformation matrix as seen in Equation (3). Since a single mode is not enough for a realistic sound, multiple instances of the mechanical oscillator can be coupled together through matrix generalization.

## 4. IMPLEMENTATION

The synthesis is implemented directly into Unity using C# utilizing Unity's base class `MonoBehaviour` to access the function `OnAudioFilterRead` that can insert data directly into the audio buffer at sample rate.

### 4.1 Exciters and Resonators

The implementation allows the user to choose which objects produce sound and select between three interaction types for those objects. Each sound-emitting object has its own individual audio source, which allows the output to be spatialized based on its position in the 3D space. The three interactions are based on the exciter-resonator relation, where one object acts as a hammer and adds the excitation to the resonating object. The first two interactions are exclusively as an exciter or resonator while the third is both of them combined.

The first interaction type is where the object only is an exciter. The exciter only emits sound when colliding with a resonator. This is useful for moving objects hitting non-moving objects, like a ball hitting a wall or rolling on the floor.

Having an object set to the second interaction type of a resonator is especially useful for static objects. These are objects that should not act as exciters themselves. In a virtual environment this can be walls, floors or other non-moving objects.

The third interaction type is an object being both an exciter and a resonator, which gives the full effect of the system. Realistically, all objects are both exciters and resonators, but this does not necessarily have to be the case in virtual environments. This is useful for moving objects that create a sound, like a falling plate or a rolling glass.

### 4.2 Material Properties

As described in Equation (1) and modal synthesis, the sound of the mechanical oscillator is a result of the normal modes of the resonator, their weighting ratios and quality factors. The normal modes can be found by analysing a sound or

computing it from object meshes alongside some of its material parameters. Any number of modes can be used for the material. More detail is obtained with the use of more modes at the cost of additional computation power. A post-gain is applied to the output amplify the signal that is otherwise a very low value. In some cases this could affect the sound, but working with the C# type floats that stores 32-bit floating-point values gives a high bit depth.

Beside the modes and its corresponding parameters, an object can also be set to be "rollable", adding the rolling sound to the object. There should be distinguished between sliding objects that require a friction sound and rolling objects.

### 4.3 Micro-Impact Rolling

The sound of rolling can in many instances be described as many small impacts [16]. How rough a surface is will have an effect on the micro-impacts. Consider rolling sound in dirt, cobblestones or smooth, new asphalt. On most surfaces, where the roughness is not entirely visible like it is with cobblestones, the rolling sound can be modeled with randomly spaced impacts. The micro-impacts must be within a maximum and minimum margin, as too far spaced apart impacts sound like a repeated knocking, while impacts too close to each other makes it sound like a continuous sound and not rolling. The random factor is important to avoiding the rolling keeping a constant frequency and evolving into what resembles a tone. Rolling can be modelled by having the impacts occurring in between durations of time where the force is set to 0.

### 4.4 Object Interaction

The triggering of sound is achieved by utilizing MonoBehaviour's collision system in the physics module. These function are called when objects enter, stay or exit a collision. Upon entering a collision, the relative velocity between the objects can be found, but the magnitude of this is not precise enough. If only this value is found, moving from one surface to another without altitude change or impacting at a narrow angle can have the same velocity as a direct impact, orthogonal to the surface. The velocity for the impact is found by taking the dot-product of the velocity and the normal vector of the contact point. If an object rolls from one surface to another without moving the object vertically, the velocity excitation to control the impact sound is zero.

If an object is set to being rollable, the velocity is found as the objects stay in a state of collision. The velocity is mapped to control the time between each micro-impact. The more micro-impacts are heard, the faster the rolling is heard to be. Once the objects exit the collision, there is no more force added to the system and the audio fades out naturally as the impacts would.

### 4.5 Modal Weight Texture

As seen in Fig. 1, a custom shader is built that can contain multiple textures. The special modal textures do not affect the visual aspect of the object as they are never rendered,

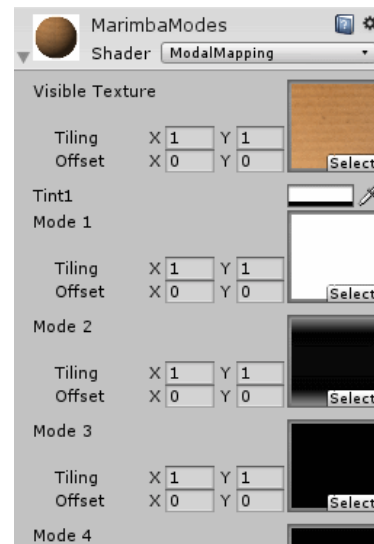


Figure 1. View of the custom shader from the inspector.

but their modal weighting values can still be accessed. The precision of the model weighting obtained depends on the resolution and interpolation of the created texture as well as how the image file of the texture is imported with or without compression. The contact point of that returns the pixel and its modal weight can either be found using the contact point provided by the collision system or as a ray-cast when the mouse is used.

## 5. OBJECT SIMULATIONS

The implementation described has been applied to three different acoustic system with impact-specific interactions. Fig. 2 shows a visual representation of the scenarios using the Unity engine. The first is marbles, that only are affected by gravity and objects in its way, while the second is a marimba instrument and the third is the surface of a glass table. Combined, these interactions show the contact synthesis in instances with no modal texture, a one-dimensional modal texture and a two-dimensional modal texture.

### 5.1 Rolling Sphere

The spring-mass system can be used for a free-falling object impacting at a single point. The total force impacting on such an exciter is given by

$$f^{(h)} = f - m_h \cdot g \quad (5)$$

where  $f$  is the impact force approximated using Newton-Rhapson and  $m_h$  is the mass and  $g$  is gravity. This results in multiple impacts like a bouncing ball, but the system determining the force can be set to zero after a single impact, thus only emitting the sound of a single impact. Similarly to Equations (2) and (3), the force of the exciter is also discretized focusing on the displacement and velocity, that is used to approximate the total force using Newton-Rhapson.



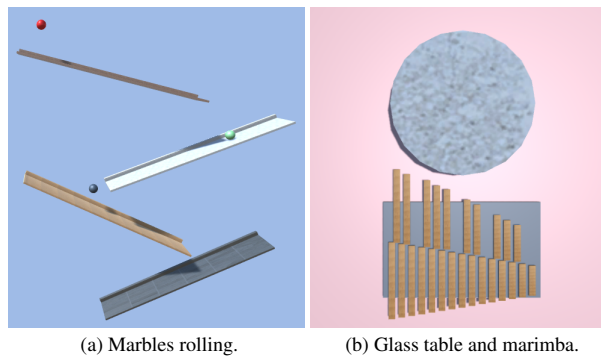


Figure 2. The interactive simulations built in the Unity game engine. Marbles are rolling down only affected by gravity on the left while specific positions can be pressed on the right hand side using the mouse.

An example using marble-like objects rolling down a track is shown on the left hand side of Fig. 2. The marbles are on “rollable” planes, as described in Section 4.4, where the material of each plane determines the characteristics of the impact and rolling. More accurate simulations can be achieved by taking the incline into account.

## 5.2 Marimba Instrument

The second physical model is the marimba instrument. The marimba is described in detail by La Favre [19]. A marimba is an idiophone made up of bars that vibrate in complex patterns resulting in the unique sound. It has been shown that up to 25 modes of vibrations ranging from 0 to 8,000 Hz can be found in a struck C3. Simple marimbas need only be tuned for the fundamental frequency, while concert marimbas will be at least “triple tuned”. The marimba can be modelled by having a fundamental frequency with the second and third modes at 4.0 and 9.2 times the fundamental [20], though some calculations put the third mode above 10.08 times that [19]. A fourth mode can be found at 19.6 times the fundamental. The modes are transverse modes, and it is shown that the amplitude of the modes change depending struck position on the bar by the mallet [21].

For this implementation, the modes and their respective amplitudes at four positions are found by La Favre [21]. The bar is struck at 4 positions; center, off-center, off-edge and at the edge of the bar. These values are one-dimensional, as the position only changes in one dimension from center to edge. A two-dimensional modal texture could be obtained by striking the bar across the shorter edge at the same positions.

The instrument is controlled using the mouse. On a click, a raycast traces a line to the hit object and obtains the UV coordinates of the raycast hit position. The UV is transformed to pixel coordinate by multiplying by the texture’s width and height and the color (and therefore modal weighting) of the pixel at the coordinate is obtained. The mechanical oscillator is excited by an initial velocity at impact.



Figure 3. Textures used for the marimba bar, with intensified brightness. The grayscale value of the color is the modal weight. The leftmost depicts the fundamental frequency while the rightmost in the connected field depicts the fourth mode. In this instance, all other position’s weighting values are scaled compared to the fundamental frequency, the modal texture of the fundamental frequency is a single value across all positions.

Fig. 3 shows the four mode textures with intensified lighting, as it otherwise would be too hard to distinguish between the dark colors. The four textures are applied to every marimba bar in Fig. 2 and once hit, the value at the point is passed on to script controlling the synthesis. To scale each position to approximately the same overall amplitude, the first mode is set to the same for each position and the rest are scaled after it. The values are set between 0 (not inclusive) and 1, with 0 being completely black and 1 being white. It is not possible to have a value of 0, as the mechanical oscillator model at one point divides by the modal weighting. To ensure a value of 0 isn’t read due to import and compression settings, a check is done while getting the pixel, setting it no less than a minimum value.

## 5.3 Circular Glass Table

The surface of a glass table is a simple everyday object with a circular plane as its top. Since the diameter is equal all around table, the amplitude of each mode is equal for all positions at equal direct length to the edge or the center. Five recordings were obtained from the glass table and used to create the modal textures. They can be described from the edge to the center as the edge, off-edge, middle, off-center and center. By analysing the recordings to see which modes were prominent across all of them, 12 modes were selected to create 12 modal textures.

The surface plot in Fig. 4 shows the values of one modal texture as height data. A circular texture of the modal weight of the recordings is created by rotating and interpolated line of the data around the center. The texture used is 800 times 800 pixels for the surface plot, but it can be as low as 9 times 9 pixels (five for the radius including the center and 9 for the whole diameter) if Unity’s own image stretching is used for the interpolation, though higher

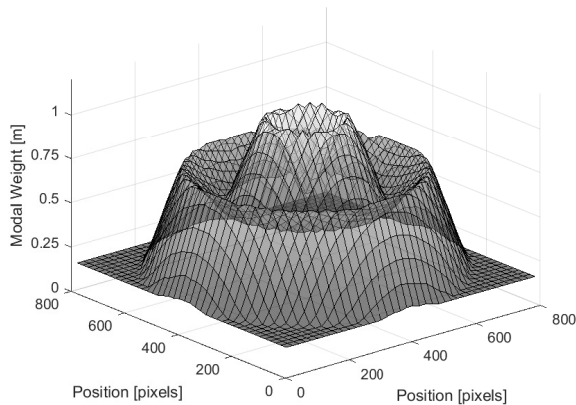


Figure 4. A 3D surface plot of the texture for the third mode of the glass table. The recordings of five positions from the center to the edge are interpolated around in a circle with the center as the anchor to create the 2-dimensional texture.

resolution textures are recommended to achieve smoother transitions between the weighting of each position.

## 6. CONCLUSION

In this paper an approach to using physical modelling for impact and rollings sounds in a virtual environment is proposed. This approach also includes the use of graphical textures for modal weightings to simulate position-dependent impacts.

The implementation can be extended to other interactions than an impact. Friction can use the same mechanical oscillator albeit with a different force excitation. The friction interaction of rubbing on a glass could use the modal texturing.

There are clear advantages to this combination of computer graphics and physical modelling of sound synthesis using the mechanical oscillator. Theoretically, if an object is modelled in 3D and a visual texture already is created, modal weights from recordings of its real-world counterpart can be specified to a point on the visual texture, and a modal texture can be computed for the whole model.

Since the modal textures are computed offline, which saves computational cost at runtime, the end result can be described as conveniently placed lookup-tables.

## 7. REFERENCES

- [1] M. Aramaki and R. Kronland-Martinet, "Analysis-synthesis of impact sounds by real-time dynamic filtering," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 2, pp. 695–705, March 2006.
- [2] K. van den Doel, P. G. Kry, and D. K. Pai, "Foleyautomatic: Physically-based sound effects for interactive simulation and animation," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, 2001, pp. 537–544. [Online]. Available: <http://doi.acm.org/10.1145/383259.383322>
- [3] J.-M. Adrien, "The missing link: Modal synthesis," in *Representations of musical signals*, 1991, pp. 269–298.
- [4] K. Van Den Doel and D. K. Pai, "Modal synthesis for vibrating objects," *Audio Anecdotes. AK Peter, Natick, MA*, pp. 1–8, 2003.
- [5] R. Michon and S. Martin, "Mesh2faust: a modal physical model generator for the faust programming language application to bell modeling," in *Proceedings of the International Computer Music Conference (ICMC-17)*, 2017.
- [6] Z. Ren, H. Yeh, and M. C. Lin, "Example-guided physically based modal sound synthesis," *ACM Trans. Graph.*, vol. 32. [Online]. Available: <http://doi.acm.org/10.1145/2421636.2421637>
- [7] S. Van Duyne and J. Smith III, "The 2-D digital waveguide mesh," in *Proceedings of the International Computer Music Conference*, 11 1993, pp. 177 – 180.
- [8] F. Avanzini, S. Serafin, and D. Rocchesso, "Interactive simulation of rigid body interaction with friction-induced sound generation," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 1073–1081, Sep. 2005.
- [9] S. Conan, E. Thoret, M. Aramaki, O. Derrien, C. Gondre, R. Kronland-Martinet, and S. Ystad, "Navigating in a space of synthesized interaction-sounds: rubbing, scratching and rolling sounds," in *16th International Conference on Digital Audio Effects (DAFx)*, Sep. 2013, pp. 202 – 209. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-00877652>
- [10] N. Bonneel, G. Drettakis, N. Tsingos, I. Viaud-Delmon, and D. James, "Fast modal sounds with scalable frequency-domain synthesis," *ACM Trans. Graph.*, vol. 27, 08 2008.
- [11] V. Zappi, A. Allen, and S. Fels, "Shader-based physical modelling for the design of massive digital musical instruments," in *NIME*, 2017.
- [12] E. Gallo and N. Tsingos, "Efficient 3D Audio Processing on the GPU," *ACM Workshop on General Purpose Computing on Graphics Processors*, ACM, Aug. 2004, poster. [Online]. Available: <https://hal.inria.fr/inria-00606754>
- [13] D. K. Pai, K. v. d. Doel, D. L. James, J. Lang, J. E. Lloyd, J. L. Richmond, and S. H. Yau, "Scanning physical interaction behavior of 3D objects," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. ACM, 2001, pp. 87–96. [Online]. Available: <http://doi.acm.org/10.1145/383259.383268>

- [14] A. MacGregor, "The sound of Grand Theft Auto V," in *Game Developers Conference*. Rockstar North, 2014, accessed: 2019-02-14. [Online]. Available: <https://www.gdcvault.com/play/1020587/The-Sound-of-Grand-Theft>
- [15] A. McLeran, "The future of audio in unreal engine," in *Game Developers Conference*. Epic Games, 2017, accessed: 2019-02-14. [Online]. Available: <https://www.unrealengine.com/en-US/events/gdc-2017-the-future-of-audio-in-unreal-engine>
- [16] D. Rocchesso and F. Fontana, "The sounding object," *IEEE Multimedia - IEEE MM*, 01 2003.
- [17] G. Borin, G. De Poli, and D. Rocchesso, "Elimination of delay-free loops in discrete-time models of nonlinear acoustic systems," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 597–605, Sep. 2000.
- [18] D. W. Marhefka and D. E. Orin, "A compliant contact model with nonlinear damping for simulation of robotic systems," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 29, no. 6, pp. 566–572, Nov 1999.
- [19] J. La Favre, "Tuning the marimba bar and resonator," <http://www.lafavre.us/tuning-marimba.htm>, 2007, accessed: 2019-02-10.
- [20] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*. Springer, 1998.
- [21] J. La Favre, "Position of mallet blow on bar - effect on bar timbre (sound quality)," <http://www.lafavre.us/FFT-mallet-position.htm>, 2007, accessed: 2019-02-10.

# Analysis of Vocal Ornamentation in Iranian Classical Music

Sepideh Shafiei

Graduate Center, City University of New York  
mshafiei@gradcenter.cuny.edu

## ABSTRACT

In this paper we study *tahrir*, a melismatic vocal ornamentation which is an essential characteristic of Persian classical music and can be compared to yodeling. It is considered the most important technique through which the vocalist can display his/her prowess. In Persian, nightingale's song is used as a metaphor for *tahrir* and sometimes for a specific type of *tahrir*. Here we examine *tahrir* through a case study. We have chosen two prominent singers of Persian classical music one contemporary and one from the twentieth century. In our analysis we have appropriated both audio recordings and transcriptions by one of the most prominent ethnomusicologists, Masudiyeh, who has worked on Music of Iran [1]. This paper is the first step towards computational modeling and recognition of different types of *tahrirs*. Here we have studied two types of *tahrirs*, mainly *nashib* and *farāz*, and their combination through three different performance samples by two prominent vocalists. More than twenty types of *tahrirs* have been identified by Iranian musicians and music theorists. We are currently working on developing a method to computationally identify these models.

## 1. INTRODUCTION

The repertoire/system of Persian classical music, *radif* consists of seven *dastgāhs* and five *āvāzes* (secondary *dastgāhs*). Each *dastgāh* consists of several pieces (*gushes*). These *gushes* are in different *maqāms* and they are related to each other through a special order, which provides a path for modulation from one *maqām* to another inside a given *dastgāh* [2]. *Radif* is a model and source for improvisation. The pieces in vocal and instrumental *radifs* are rarely performed exactly as they appear in *radifs*. The musicians use the models and patterns in *radif* to improvise new pieces. During the twentieth century, the *radif* was established as an icon of tradition, authenticity, and heritage. It has been the center of discourses about preservation, change, creativity, imitation, individuality, emotion, style, meaning, authority, and national roots in Iranian music. Through these discourses, the *radif* has been developed as a two-headed arrow pointing towards the future and creativity, and at the same time towards the past and authenticity.

There are two main recorded vocal *radifs* sung by two masters of the art during the twentieth century: Davāmi and Karimi.

*Tahrir* is rapid transition between the main note and a higher-pitched note. The second note is usually referred to as *tekyeh*, which means leaning in Persian. From signal processing perspective one of the differences between *tahrir* and vibrato is that the pitch rises and fall in *tahrir* is usually sharper and the deviation from the main notes can be larger compared to vibrato [3]. Also, the oscillation in vibrato is toward both higher and lower frequency around the main note, but in *tahrir* mainly the higher frequency is touched abruptly. There are different types of *tahrir* in Persian vocal music that can be categorized from both performance style perspective and from studying the melodic contour.

We have decided to study the transcriptions of *radif* as well as the audio, since these transcriptions are among the main sources for teaching and learning *radif*. Musical notation has a long history in Iran. We can see early examples of musical notation in Maraghi's works in 14th century [4]. He uses alphabet letters to show the pitch and rhythmic circles to illustrate the rhythm of the pieces. When western musical notation was introduced in Iran, it naturally replaced the use of alphabets and rhythmic circles [5]. Nowadays it is part of the musical pedagogy in modern Iran. Furthermore, transcriptions of vocal *radifs* are among important sources for instrumentalists who usually accompany the vocalists in a form of *āvāz* and *javāb āvāz* (question and answer). In this form the vocalist sings a hemistich of poetry and the instrumentalist plays a short sentence as a reply to that. The musical intervals we see in the transcriptions in this paper, although in some cases are different than what the vocalists sings, are the intervals that instrumentalists use in their answer to the voice.

In order to understand different types and styles of *tahrir* we need to parametrize the characteristics of *tahrir*. Since there has not been enough computational models for analyzing *tahrir*, the parameters of vibrato can be a good start for modeling different types of vocal embellishments. Luwei mentions four computational attributes for vibrato: rate, extent, sinusoidal similarity, and envelope. Vibrato rate determines the tempo of the vibrato, the extent shows the variation in the fundamental frequency of the pitch in vibrato, sinusoidal similarity examined the similarities between the shapes of vibratos, and envelope which shows the changes in the vibrato extent [6].

Copyright: © 2019 Sepideh Shafiei. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



## 2. VOCAL TRADITIONAL MUSIC IN IRAN: A BACKGROUND

### 2.1 A brief history of musicological research and music education in modern Iran

Interest in education and research on Persian music increased during the twentieth century. Among the most prominent musicians from the early twentieth century to the 1970s, one can mention Vaziri and Khāleqi, who were both modernist and in favor of using Western music methods and ideas to “improve” Persian classical music. The efforts of such modernist musicians changed the status of music in the society. The classical Iranian music became recognized as an element of “high culture” associated with the newly formed urban middle class. Western ethnomusicologists started to visit Iran for fieldwork and ethnographic research. The main areas of their focus were Persian classical music and folk music. Among the prominent Western ethnomusicologists who worked on Persian classical and folk music, one can mention Nettl [7] and Blum [8] and [9], who did their fieldwork in Tehran and Mashhad in the 1960s, Zonis who visited Tehran during the years 1963-1965 [10] and [11], and During who visited Iran multiple times since the mid-1960s. They all worked closely with very prominent musicians of the time in Tehran and other large cities. The first works on Persian classical music were mainly devoted to different aspects of the *radif* [12], [7], and [13], as well as biographies of musicians [14] and [15], documentation, transcription, and archiving [1].

The process of documentation and transcription of the *radif*, together with the availability of recording technology, partially, implicitly, and gradually changed the music scene of Iran. The idea of preservation and protection started to work, to some extent, against itself, even before scholars could notice the flaws and the contradictions of this idea. Descriptive transcriptions of the *radif* by various Iranian and Western ethnomusicologists and music scholars, and the recordings of masters, later served as sources of knowledge. The practice of *radif* has changed partially from an oral tradition to a written tradition. The role of memorization of the whole *radif* has been reduced to a great extent. Students learn “improvisation” more as a technique, and perhaps to some extent mechanical, rather than as a result of full and in-depth knowledge of *radif*. Many students use recordings of different masters and transcriptions of *radif* to familiarize themselves with various performing styles. The direct master to student teaching, which was historically central to the practice of the *radif*, became inevitably less important in the new setting. This “modern” setting brings up many questions regarding the forms of continuity and discontinuity in the functions and directions of traditional music in today’s Iran.

### 2.2 Traditional Music after the 1979 revolution

Historical events<sup>1</sup>, after the 1979 revolution and the anti-Westernization movement changed the cultural scene of Iran. The restrictive cultural policies of the government almost eliminated production of popular music. The government defined the “appropriate” (*mojāz*, acceptable) forms of music, whose definition always remained vague and changing. The lyrics have been among the important elements for deciding the “appropriateness” of music. Persian classical music, traditionally, has been linked to masterpieces of Persian poetry, such as *ghazals*<sup>2</sup> of Hafez, Sa’di, and Molavi (Rumi). This is an important factor that gives traditional music a relatively safe position. Another factor in deciding on the “appropriateness” of the music is the performers. The government accepts older male musicians more easily compared with their young and/or female counterparts. In general, there are always exceptions to these rules. Because of the nature of traditional music, it has always been one of very few genres that is judged as “appropriate.” In the absence of popular music, famous traditional musicians, such as Shajariān, Lotfi, and Alizādeh gained the social popularity of pop stars. This made the prominent traditional musicians less accessible for teaching. Many of these musicians no longer accept beginning students. Many of them teach workshops that accept a limited number of performers from many applicants. These social factors contributed to fundamental changes in the classical music.

After the 1979 revolution, international policies made Iran a difficult destination for Western visitors, including ethnomusicologists. Furthermore, because of governmental censorship, the safest areas of studies for insider scholars were those that did not involve any social and political issues. Hence (purely) musicological study of “appropriate forms of music” has been one of the most popular topics for Iranian ethnomusicologists after the revolution. Among the more recent works in this field one can mention Bubān’s dissertation which compares the rhythmic patterns of the Persian language with rhythmic patterns of the *radif* [5]. She also talks about the insufficiency of Western musical notation for rhythm in Persian music and suggests a visual notation. There are many other recent works on the *radif*, among which one can mention Asadi’s dissertation, which is on the structure of the *radif* [16], Āzādehfār’s book on rhythm in Persian āvāz [17], Mehrāni’s three-volume work on the theory of Iranian music [18], Fereyduni’s book on the characteristics of the vocal *radif* of Davāmi [19], and Jafarzādeh’s book on Iranian musicology [20].

### 2.3 Vocal Traditional Music

The word *āvāz* has several meanings in Persian. It refers to humans’ singing as well as the sound of birds and instruments in old Persian literature. In Iranian traditional music, *āvāz* specifically means the elaborate improvisatory non-metric part of the vocal performance usually accompanied by one instrument at a time in the form of *āvāz* and *javab-e*

<sup>1</sup> Among the important events one can mention the Iran-Iraq war (1980-1988), and the Cultural Revolution (1980-1983).

<sup>2</sup> A classic form of Persian poetry

*āvāz*, also known as question and answer, which is considered a dialogue between the vocalist and the instrumentalist. In this part the vocalist leads the performance and sings some or all lines of a *ghazal*<sup>3</sup>, and the instrumentalist answers creatively. The vocalist usually sings one or two verse(s) in each selected *gusheh* of a *dastgāh*.

The most common format of the performance is to start with vocables and then to sing the first line of verse in *darāmād* which is the beginning *gusheh* of each *dastgāh* and then sing other lines of verse in a different *gushehs* of the same *dastgāh* in a conventional order. Usually there is at least one main modulation, which gives a feeling of a different *maqām* and then finally last line of verse is sung in *forūd*, which is a return to the main *maqām*. The duration of *āvāz* depends on the proficiency of the singer. One of the main elements of *āvāz*, which shows proficiency in singing traditional Iranian music is *tahrir*. The expertise and the level of proficiency of a singer is evaluated mainly in this part of the performance (*āvāz*). There are singers who can only sing *tasnifs* (a metric pre-composed piece). *Tahrir* usually appears towards the end of hemistich, or on the words where vocalist want to emphasis on the meaning.

### 3. TAHRIR: A CASE STUDY

#### 3.1 Different types of *tahrir*

Mohammad Reza Lotfi, one of the most prominent Iranian musicians and *tār* players of the late twentieth century identifies seven types of *tahrir* based on Davami's performance of *radif* [19]. We studied three references in Persian that classify different types of *tahrir* [19] and [18]. *Nashib* and *farāz* are two types of *tahrirs* according to these sources.

#### 3.2 Karimi's Vocal Radif

Karimi is one of the main masters of the art in the twentieth century. His repertoire consists of 145 *gushes*. His performance is recorded and available to public. It has also been transcribed by one the most prominent ethnomusicologists, Masudiye [1]. It is later transcribed by two other musicians, Atrāyi and Tahmāsbī. Hence for each *gushe* of Karimi's vocal *radif*, we could have three MIDI files that are slightly different. Finally, after much consideration we found Masudiye's notation more appropriate for the purpose of our study. Figure 1 shows the way we organize our study.

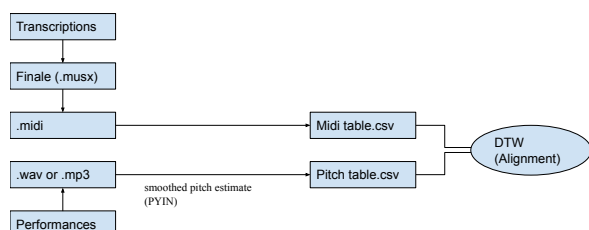


Figure 1. Audio and midi processing steps

<sup>3</sup> Other forms of poetry are also used but are not as common as *ghazal*

As can be seen in Figure 1, we have used PYIN for pitch recognition, using Sonic Visualiser, Smoothed Pitch Track transform by Mathias Mauch and Simon Dixon [21]. Parallel to the audio we have made a table corresponding to the MIDI file, and then we have used Dynamic Time Warping [22] algorithm in MATLAB to compare these two curves. We modified the MATLAB *dtw* plot function, so that we can mark the differences between the two curves. The results can be seen in Figures 2 and 4.

#### 3.3 Tahrir-e Nashib and Farāz in vocal *radif* of Karimi

*Tahrir-e nashib* (literally: descend), and *farāz* (literally: ascend) are two types of *tahrir* that is discussed by Fereyduni, Mehrani, and Lotfi. Their melodic movement as it can be inferred from their names is a slow descend or ascend towards the main note, where the vocalist or instrumentalist usually spends a relatively longer time. The movement is most of the time towards the *shāhed* or *ist*, or *owj*, which are the main functional notes in each *gusheh*. According to Owen Wright “*Shāhed* (‘witness’) is the most prominent pitch of the *gusheh*, its salience marked primarily by relative duration; *ist* (‘stand’) is an intermediate phrase final note other than the *shāhed*.” [19] p. 33. *Owj* (‘peak’) is usually a fifth above the *shāhed* of a *gusheh*.

Figure 2 shows *tahrir-e nashib* in the final phrase (*forud*) of the *gushe-ye daramād* of shur in Karimi's *radif*. The vertical axis shows the pitch value in cents and the horizontal axis is time. As we can see in this figure there is a mis-match between the audio and transcription. We have marked the duration mismatches in Midi with yellow. The red color shows the audio and blue shows the midi. Figure 3 shows the original Masudiye's transcription of the same *tahrir*. The circles below the notes show leaning (*tekyeh*) of the main note towards the higher note.

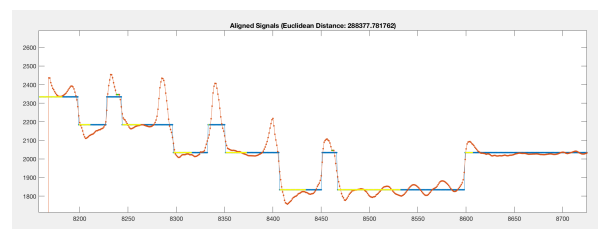


Figure 2. *Tahrir-e nashib* in *daramād* of Shur of Karimi



Figure 3. *Tahrir-e nashib* in *daramād* of Shur of Karimi, Masudiye's transcription, page 13, line 4 of *darāmād*

One of the melodic characteristics of *tahrir*, as can be seen in the above figure, is a repetition of a simpler form

or group of notes. In the above *tahrir* the repeating form consists of a note which leans toward a higher pitch. In this example the interval between the main note and the peak of the higher note is at most as high as about a tone and half ( $\approx 300$  cents). The vocalist repeats the same pattern in a descending manner. Sometimes different types of *tahrir* can be combined to form a more complicated melodic phrase. For example in *darāmad* of *bayāt tork* we have a longer pattern which consists of a *nashib tahrir* followed by *tahrir-e farāz*. As can be seen in Figure 4, the whole longer pattern is repeated twice. In figure 5 we see Masudiye's transcription of this *tahrir*.

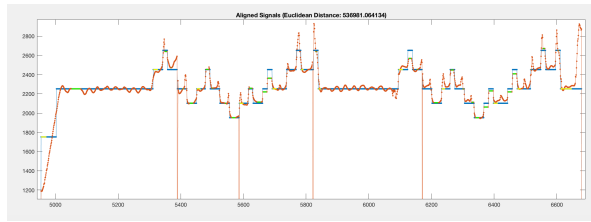


Figure 4. *Tahrir-e nashib* followed by *tahrir-e farāz* in *daramād* of Bayāt-tork



Figure 5. *Tahrir-e nashib* in *daramād* of Bayāt-tork of Karimi, Masudiye's transcription, page 49, lines 3 and 4 of *darāmad*

### 3.4 Tahrir-e Nashib and Farāz in Shajarian's Performance

Figure 6 shows a sample of Shajarian's *tahrir* in *gushe-ye Owj* in the hemistich (36':01"- 36':14'): "*baske shostim be khunābe jegar jāme-ye jān*."<sup>4</sup> This *tahrir* is on the last word, *jān*, and on vowel *ā* for 5 seconds, involving the sequence G, F, F, E, E, D, E, E, F, F, G with *tekyehs* to higher pitches. Fereyduni mentions the name "*nashib o farāz*" for this type of *tahrir* ([19], P. 19). This name is also mentioned by Payvar, in his transcription of Davami's *radif*. In this *tahrir*, the average duration to reach the peak of the *tekyeh* note from the main notes is 0.75 m.s. The highest frequency jumps in this *tahrir* are about one and half tone ( $\approx 290$  cents), and the lowest frequency jumps are about a half tone ( $\approx 90$  cents).

## 4. FUTURE DIRECTION

Our goal is to computationally analyze more *tahrir* types and their subtle differences. We would like to study *tahrirs* performed by various vocalists and to find their stylistic features.

<sup>4</sup> Hamnavā bā Bam ["Compassion for Bam"]. Delāwāz. (Tehran concert and background documentary). 2006, available at: <https://www.youtube.com/watch?v=7xalZQOFW88>

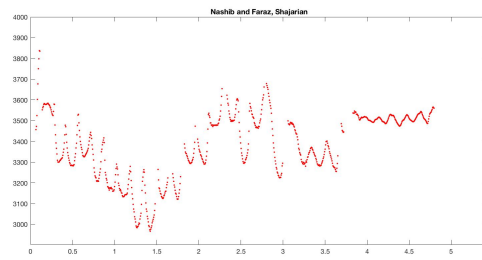


Figure 6. *Tahrir-e nashib* and *farāz* in *owj* of Bayāt-kord performed by Shajarian

## 5. REFERENCES

- [1] M. Masudiye, *Radif-e Āvāzi Mahmud Karimi*. Mahoor, 1997.
- [2] H. Alizadeh, *Theory of Iranian music*. Mahoor, 2009.
- [3] P. Bahadoran, "Analysis of tahreer in iranian traditional singing," in *Proc. Int. workshop in Folk Music Analysis*, Dublin, 2016, pp. 92–95.
- [4] A. Maraghi, *Jāme al-alhān*, B. Khazrāyi, Ed. Farhangestān-e honar, 2009.
- [5] N. Bouban, "Comparative study of rhythm in iranian music and persian language," 2009.
- [6] L. Yang, "Computational modelling and analysis of vibrato and portamento in expressive music performance," 2017.
- [7] B. Nettl, *The radif of Persian Music*. Champaign, IL: Elephant Cat, 1992.
- [8] S. Blum, "Changing roles of performers in meshhed and bojnurd iran," in *Eight Urban Musical Cultures*, B. Nettl, Ed. Urbana: University of Illinois Press, 1978.
- [9] —, "Compelling reasons to sing: The music of taziye," *TDR*, vol. 49(4), pp. 86–90.
- [10] E. Zonis, "Contemporary art music in persia," *The Musical Quarterly*, vol. 51(4), pp. 636–648.
- [11] —, *Persian Classical Music: An Introduction*. Harvard University Press, 1973.
- [12] G. Tsuge, "Rhythmic aspects of the Āvāz in persian music," *Ethnomusicology*, vol. 14(2), pp. 205–227.
- [13] H. Farhat, *The Dastgāh Concept in Persian Music*. Cambridge University Press, 1990.
- [14] B. Nettl. (1989) Borumand, nur-ali in encyclopedia iranica. [Online]. Available: <http://www.iranicaonline.org/articles/borumand-nur-ali-b>
- [15] D. Safvat. (1994) Dawāmi, abd-allah in encyclopedia iranica. [Online]. Available: <http://www.iranicaonline.org/articles/dawami>

- [16] H. Asadi, “The concept and structure of the dastgāh in persian classical music: Comparative analysis of the radif,” 2006.
- [17] M. Azadefar, *Rhythmic Structure in Iranian Music*. Tehran Art University Press, 2004.
- [18] H. Mehrāni, *Iranian Solfège Method*. Mehrāni, 2010.
- [19] N. Fereyduni, *Analysis of the Vocal Radif of Traditional Iranian Music According to the Performance of Abdollāh Davāmi*. Pārt, 2004.
- [20] K. Jafarzadeh, *Iranian Musicology*. Honar-e Musiqi, 2013.
- [21] M. Mauch and S. Dixon, “pyin: A fundamental frequency estimator using probabilistic threshold distributions,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 659–663.
- [22] D. J. Berndt and J. Clifford, “Using dynamic time warping to find patterns in time series,” in *KDD workshop*, vol. 10, no. 16. Seattle, WA, 1994, pp. 359–370.



# VUSAA: AN AUGMENTED REALITY MOBILE APP FOR URBAN SOUNDWALKS

**Josué Moreno**

University of the Arts  
Helsinki

jmo14618@uniarts.fi

**Vesa Norilo**

University of the Arts  
Helsinki

vno11100@uniarts.fi

## ABSTRACT

This paper presents VUSAA, an augmented reality soundwalking application for Apple iOS Devices. The application is based on the idea of Urban Sonic Acupuncture, providing site-aware generative audio content aligned with the present sonic environment. The sound-generating algorithm was implemented in Kronos, a declarative programming language for musical signal processing. We discuss the conceptual framework and implementation of the application, along with the practical considerations of deploying it via a commercial platform. We present results from a number of soundwalks so far organized and outline an approach to develop new models for urban dwelling.

## 1. INTRODUCTION

City dwellers tend to be in the public space only in their transits, going from home to work to groceries to recreation and back home. The faster the transit time the better, occupying public spaces only while consuming. The use of headphones while transiting fosters new forms of urban detachment, this ‘headphone city’ [1] alienates any sense of place or community links. We believe that sound and the practice of soundwalking are very powerful tools to create sense of place and can lead to new forms of urban dwelling.

In this paper, we introduce the *Virtual Urban Sonic Acupuncture App (VUSAA)*, a novel iOS application that generates site aware augmented reality urban soundwalks. The app relies on the practice and concepts behind *sonic acupuncture* and *aural weather* that Moreno is developing in his research [2]. *Urban Sonic Acupuncture* is the parallel in the sonic field of the hyper-local urban rehabilitation practices under the name of *urban acupuncture* [3] [4]. *Aural weather* echoes prior theoretical work in design and architectural atmospheres [5] [6]. The concept of “weather” also refers to alternative ways of organizing sound [7].

VUSAA is an experimental augmented reality soundwalking app for iOS mobile devices, based on *urban sonic acupuncture* strategies. It uses sensor technology to generate sonic content intended to affect the perception of the urban environment. VUSAA constitutes a proposal to use soundwalking and urban psychogeographic drifting as tools

for finding new ways of urban attachment and conscious dwelling by using minute sonic interventions that augment the sonic reality of the urban space.

The rest of the paper is organized as follows: Background (Section 2) gives an overview of the prior work on soundwalk applications and related theory. The Conceptual Framework is discussed in Section 3. We follow up with a description of the Implementation and the achieved results in Sections 4 and 5. Conclusions are presented in Section 6.

## 2. BACKGROUND

In this section we discuss the concepts and practices that helped in building the conceptual and technical background of the augmented reality soundwalking app VUSAA.

### 2.1 Urban Sonic Acupuncture

To address the issue of urban decay, recent urban planning trends opt for small-scale local participatory actions that have a global impact in urban life and urban stewardship. This practice has been called *urban acupuncture* [3] [4]. *Urban sonic acupuncture* offers a sonic take on *urban acupuncture* practices, based on cultural acoustics and aural architecture [8]. Urban sound, its auditory figures [9], sound effects [10] [11] and *sonic commons* [12] have been thoroughly studied.

Using sound in this way foment urban attachment, social dialogue, different speeds and wandering [13] in public spaces. VUSAA offers the possibility of testing the idea of *urban sonic acupuncture* in the form of augmented reality, generating minute subtle sonic content according to the present sonic situation and other environmental data gathered by the sensors on a mobile device. These sonic interventions affect the perception and relationships of the sounds and events that the listener participates in.

In developing the practice of *sonic acupuncture*, the concept of *aural weather* was coined by Moreno [2] and formulated inspired by the theory of atmospheres [5]. This concept can also refer to alternative ways of organizing music [14] [7], as shown in Section 3.1.

### 2.2 Prior Soundwalking Apps

Some augmented reality and soundwalking apps exist in the iOS App Store. *GeoComposer/GeoPlayer* from *sonic-Planet* links field recordings and pre-composed sonic content to the Google Street View. This creates site-specific 3D

Copyright: © 2019 Josué Moreno et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

soundscapes. The soundscapes can be experienced on-site as audiovisual augmented reality or remotely.

*Collectif MU* has developed *SOUNDWAYS*. This app plays back pre-composed music which gets triggered when the user enters a specific area linked to a GPS position. The app displays the area of playback for each sonic event as bubbles of different size and loudness–balance behaviour. The walks can be done on-site or remotely, by moving a cursor around a map on which the bubbles are displayed. Based on this platform, *MSWalks* app was developed for the Sibelius Academy Music Technology Department. The *MSWalks* app contains soundwalks specific to the Helsinki Töölönlahti area. There are plans to include more soundwalks for different areas in Helsinki.

*Walk With Me* by Strijbos & Van Rijswijk is another app with similar GPS-linked sonic bubbles, with the addition of text information as means of extending the experience of visiting places. Significantly, this app uses the microphone input and site-specific sound effects to modify ambient sounds.<sup>1</sup>

### 2.3 The Kronos Programming Language

Kronos is a declarative programming language designed for musical signal processing. It features declarative, generic programming and a semi-functional reactive model. [15] There is also prior work on deploying Kronos programs on mobile devices [16].

Kronos applications are modeled as reactive signal graphs. Implementing an augmented reality application consists of connecting the various sensors and inputs on the physical device to the signal graph, which in turn produces the result audio stream.

## 3. CONCEPTUAL FRAMEWORK

In all the apps mentioned in Section 2.2, soundwalks are bound to specific geographical areas and the sonic content is preordained. None of them uses real-time generated music; they rely on pre-composed or pre-recorded material. Among them, only *Walk With Me* utilizes real-time audio input in any way.

A motivator for the present study was our belief that using the microphone input and listening to it, is what makes the actual environment and its inherent sound relevant to the provided sonic content. *VUSAA* proposes a generative way of engaging sonically with the environment. This is achieved by making the app aware of the present sonic and other site-related conditions. Therefore, we decided to exclusively support on-site, augmented reality soundwalking.

The work of the duo A+O (Sam Auinger and Bruce Odland) “*Harmonic Bridge*” (MASS MoCA, North Adams, MA, 1997–present), Max Neuhaus “Times square”, and Alvin Lucier’s slow sweeping waves pieces and “*I am sitting in a room*” have played an important role in the development of sonic acupuncture strategies used in *VUSAA*. What all these works have in common is that they take a non-exclusionary, transformative attitude towards diegetic

sound. This resonates strongly with our goal of urban sound augmentation. In addition, *VUSAA* takes direct inspiration from the practice of *soundwalking* [17] and *psychogeography*’s [18] inclination to drifting.

### 3.1 Composing Aural Weather

When composing for an unknown range of possibilities, spaces and local conditions, it is important to find the balance between the variety of generated sonic reactions and the identity of the work itself. In order to compose *aural weather*, one has to learn how to create music with a primary dimension other than *time*, learning “*to move from structure to process, from music as an object having parts, to music without beginning, middle, or end, music as weather*” [14]. The challenge consists of creating the conditions in which the sonic atmosphere (*aural weather*) provided by *VUSAA*, installs itself in whatever pre-existing sonic atmosphere the user might find.

The theory of atmospheres has gained momentum recently in the fields of architecture and design [5] [6] [19]. These practices foment peripheral perception and light Gestalt as means of approaching architecture and design. To install an atmosphere, Thibaud suggests learning how to master “*the art of transpiring, the art of coloring, and the art of accompanying*” [19]. The connotations of weather, that atmospheres already have, are suggested in sonic context by Ingold [20]. Along the same lines, the concept of acoustemology – knowing a place through sound [21] – is highly relevant.

From all this, we envisioned a reactive multi-layered “weather” in the form of an algorithm with no pre-recorded material. The algorithm is designed to allow for driving the user’s attention towards surprising elements in the urban soundscape. In this way, the *aural weather* behaves dynamically and musically under complex sonic situations such as heavy traffic. On the other hand, it gets very subtle under more a tranquil atmosphere.

Finally, a musical structure that is not based on time requires a great deal of user agency in articulating the temporal dimension, choosing the walking path, the walking pace, choosing where to point the device’s microphone at, etc. We tried to avoid the need to interact with the screen. However, we added a slider to the main screen to let the user adjust their preferred listening balance between the microphone and generated sound.

## 4. IMPLEMENTATION

This section discusses the implementation of the iOS application supporting the soundwalk concept that is the topic of the present study. We will describe the user interface and sound processors in *VUSAA* as well as some details specific to deploying the application on iOS.

### 4.1 Presentation

The user interface in *VUSAA* is meant to divert the user from the screen, prioritizing other means of agency, such as speed of walking, choice of path, or where to point the device’s microphone.

<sup>1</sup> All the apps mentioned in this section can be downloaded from the iOS App Store or directly from the embedded link in the text.

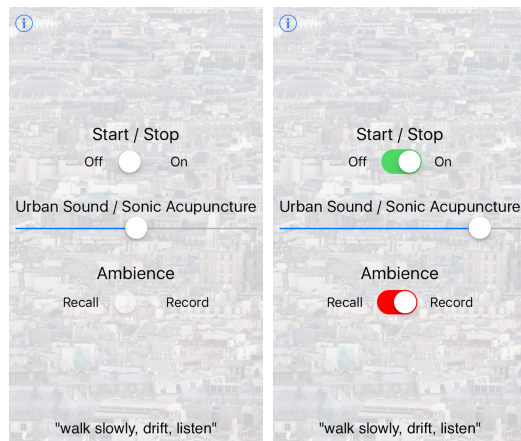


Figure 1. On/Off VUSAA main screen.

The on-screen user interface consists of a slider and two buttons; please see Figure 1. The application features an ambience loop of 5 seconds. The user may toggle between recording and reproduction of the loop by toggling the *Ambience* switch; when record is engaged, the audio loop is overdubbed. The recorded urban sound will be subsequently played back at different times and loudness levels.

The rest of the interface consists of the info screen and the statement *walk slowly, drift, listen*. In the info screen, there is information about how to use the app and a small introduction to the concepts behind *urban sonic acupuncture* and VUSAA. We understood that such an explanation would be needed to make the app and its sparse user interface easy to navigate.

## 4.2 Sound Generation

The signal flow diagram of the VUSAA application is shown in Figure 2. The actual signal processor responsible for sound generation is implemented in the Kronos programming language.

The application derives control data from transient analysis, as well as non-audio inputs including luminance (via camera), GPS position and the time of day relative to local sunrise and sunset.

The control data is used to drive several synthesis algorithms in parallel. The recorded ambient loop is processed with a luminance-dependent high frequency rolloff and mixed with a thresholded noise generator (“dust”) and fed into a resonator bank. The bank is tuned to a chord that changes with the time of day.

Another preset chord controls a pseudorandomly arpeggiated Karplus-Strong string model with a luminance-dependant damping parameter. The excitation for the model is derived from the microphone input, gated by the transient detector; percussive sounds in the ambience gain a windchime-like echo in the augmented soundscape.

A progressively transposing echo is implemented with a tape loop -style delay effect with a real-time write head and a slower read head. The transient detector causes the heads to realign, giving the effect of resetting the transposition when an audible echo begins.

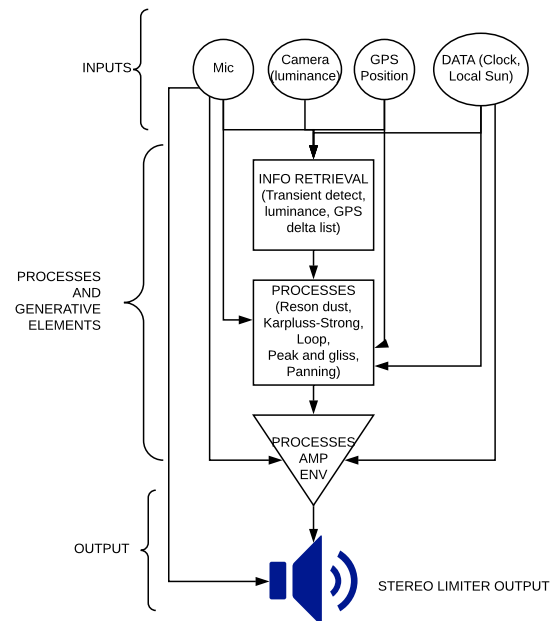


Figure 2. Overview of the VUSAA algorithm.

All the processes described so far are modulated by slow, gradually phasing amplitude envelopes. The overall speed of the envelope is determined by time of day, while the least significant portion of GPS latitude and longitude change the phase offsets between the envelopes. This causes new aspects of the mix to appear as the soundwalker moves around.

Finally, the synthesized sound is mixed with raw microphone input according to the slider position on the user interface.

### 4.2.1 iOS Deployment

Kronos is originally a just-in-time compiler. A mobile application should, however, be statically compiled. Apple guidelines prohibit code generation on the device, and in any case it is best to take that burden away from the end user.

Kronos does also feature a static compiler called *kc*, incorporating the LLVM [22] code generator. As such, it can generate C-compatible object files, LLVM bitcode, or symbolic assembly. LLVM is also inherently a *cross-compiler*, which means that it can, by default, generate code for architectures other than the host machine it is running on. For iOS development, we configured and built LLVM and Kronos on macOS with support for x86 and ARM.

When compiling for iOS, Xcode behaves differently depending on the target. If a physical device is connected, the code generators target its native hardware architecture. For the iOS simulator, x86 is used. When deploying to the app store, Xcode builds code for all the three variants of ARM processor currently supported. As of this writing, the Apple App store can also record binaries in the LLVM bitcode. These attain a degree of hardware independence, and in theory allow support for as-of-yet unknown future



architectures.

#### 4.2.2 Xcode Integration

We integrated *kc* into Xcode with a custom build rule. This way, the Kronos-language DSP implementation can be added to the Xcode project as a normal source file. Xcode automatically sets a *TARGET\_TRIPLE* variable describing the platform and architecture for each build. The build rule relays that to the *kc* driver, and thus to LLVM, resulting in transparent cross compilation support; Xcode invokes *kc* to build the DSP code for any required hardware targets.

Because we used a non-standard, cross-compilation capable build of *kc* for the present project, the specific compiler binaries were embedded and archived within the *VUSAA* repository.

## 5. RESULTS AND DISCUSSION

### 5.1 Public Presentations

We presented *VUSAA* in two events, involving a short theoretical presentation and a soundwalk utilizing the app. Given that not all the attendants had an iOS device or headphones, devices were provided along with headphones. Headphone splitters were used to stretch the number of available iOS devices, allowing for a social soundwalking experience in pairs, as shown in Figure 3. Having two users per device meant that two people would have the same sonic experience, having to negotiate the path to follow and the app settings.

The first event happened in Venice in June 2017 in the Research Pavilion – Camino Events of the University of the Arts Helsinki. The *VUSAA*-enhanced soundwalk was hosted in the Giudecca island in the surroundings of the pavilion. This was a very interesting place for soundwalking given the absence of car traffic and the interesting noises produced by the vaporetto platforms. A group of international researchers contributed to the discussion after the soundwalk.

The second event happened in the Tampere Biennale in Finland for several days. On the first day, we used the app to transit between art galleries in the opening day. That day, some sound artists started spontaneously using the app to listen to other sound art works and musical compositions. For the second soundwalking event, the walk happened around the centre of the city, designed by the production team to experience the app in different urban settings that ranged from a waterfall area to the transit under a heavily-used car bridge.

We toured intensively during the Summer of 2017, testing the app in multiple countries. As a result, we produced a set of recordings documenting the app evolution and reactions to different urban environments. We presented these recordings and the concepts behind *VUSAA* at the Sibelius Academy Music Technology Department's *Generative Art Café* in Helsinki.

We gathered feedback from the users in the form of notes taken during discussions following each presentation and during personal interviews. It seems that *VUSAA* is particularly successful in creating relationships among pre-existing



Figure 3. *VUSAA*-enhanced second soundwalking event during Tampere Biennale 2018.

sound objects. For instance, the always moving “dust” element is a subtle and yet powerful way of affecting the perceived fundamental of pitched sound objects. Simultaneously, the resonant algorithms are efficient in generating harmonic content and creating harmonic relations between disparate sound objects.

### 5.2 Problems with the App Store review process

We faced an interesting problem during the app review and publishing process. *VUSAA* was originally meant to work in background mode to discourage users from staring at the screen only. The iOS App Store review team initially rejected our app because they did not find a compelling reason for the application to be accessing the microphone while in the background. After some back and forth submission-rejection cycles we asked them for a phone conversation in which it became clear that their and our notion of *audio content* were not aligned. They clearly stated that resonant dust or softly playing background chords do not count as audio content.

At that point, it became obvious that our digital platforms are products subject to corporate ownership and control, and not public services. A privately owned platform has no space for flexibility in their interpretations and they have all the right to reject our submissions or even ban us from App Store. We learned that proposing an audio application that consists mainly of a complex passive sonic object was a difficult proposition for the App Store review team. We solved this by giving up the background mode altogether. This decision turned out to have no significant impact on the app usability during the soundwalking events.

*VUSAA* app can be downloaded from the App Store from this [link](https://itunes.apple.com/es/app/vusaa/id1244860977?mt=8)<sup>2</sup> or from [Moreno's website](http://josuemoreno.eu/project/vusaa/).

### 5.3 Future Work

Taking into account the received feedback and our findings through daily use, we have plans for further developing *VUSAA*'s variety of processing modules extending the possible *aural weather* situations provided by the app. We will work on keeping the app updated and extend the hardware compatibility with different headsets. At the same time,

<sup>2</sup> *VUSAA* can be downloaded from this link: <https://itunes.apple.com/es/app/vusaa/id1244860977?mt=8> ; or from the project's website: <http://josuemoreno.eu/project/vusaa/>



more VUSAA-enhanced soundwalking events are currently under production in several European cities. Finally, further development will be done in refining and implementing the interaction between control parameters and the audio algorithm.

Currently, there are new projects under development that are based on the technology and conceptual framework developed for VUSAA. These elements will allow us to implement site-specific *urban sonic acupuncture* projects quickly, by taking advantage of the rapid development enabled by Kronos.

## 6. CONCLUSION

VUSAA constitutes a first attempt at augmented reality soundwalking based on *urban sonic acupuncture* strategies for iOS mobile devices. It takes advantages of the many sensing and listening abilities the iOS mobile devices provide to generate sonic content meant to affect the perception of the urban environment.

We believe that VUSAA breaks new ground in several ways; it is the first Kronos-based application that was successfully shipped via the iOS App Store. It is arguably the first site-aware soundwalk app, and definitely the first Urban Sonic Acupuncture app available. The initial group VUSAA-enhanced soundwalks and the received feedback from the users positions this app as a very interesting proposal for turning soundwalking and urban psychogeographic drifting into tools for finding new urban dwelling models in which to develop a more conscious attitude towards sound. We consider it very promising to use the very same tools that contribute to urban aural detachment – mobile devices and headphones – to revert the situation.

## Acknowledgments

Josué Moreno's work was supported by the Sibelius Academy MuTri Doctoral School, the Centre for Music & Technology and the Kone Foundation. Vesa Norilo's work was supported by the Academy of Finland, award number SA311535.

## 7. REFERENCES

- [1] C. Ripley, "Instrumental operations in the urban assemblage," in *Journal of Sonic Studies*, 11, 2016.
- [2] O. Lähdeoja, J. Moreno, and D. Malpica, "In situ: Sonic greenhouse. composing for the intersections between the sonic and the built," *Journal for Artistic Research*, 2019.
- [3] J. Lerner, *Urban Acupuncture: Celebrating Pinpricks of Change that Enrich City Life*. Washington: Island Press, 2014.
- [4] H. Casanova and J. Hernandez, *Public Space Acupuncture: Strategies and Interventions for Activating City Life*. Actar, 2015.
- [5] T. Griffero, *Atmospheres: Aesthetics of Emotional Spaces*. Routledge, 2014.
- [6] G. Böhme, "Urban atmospheres; charting new directions for architecture and urban planning," in *Architectural Atmospheres: On the Experience and Politics of Architecture*, ed. Christian Borch, Birkhäuser Basel, 2014.
- [7] D. Toop, *Haunted Weather: Music, Silence and Memory*. Five Star Fiction S., 2006.
- [8] B. Blesser and L.-R. Salter, *Spaces speak, are you listening?: Experiencing aural architecture*. Cambridge, Mass: MIT Press., 2007.
- [9] B. LaBelle, *Acoustic Territories: Sound Culture and Everyday Life*. New York: Continuum, 2010.
- [10] J.-F. Augoyard and e. H. Torgue, *Sonic Experience: A Guide to Everyday Sounds*. McGill-Queens University Press, 2005.
- [11] P. Amphoux, *Aux coutes de la ville : la qualit sonore des espaces publics europens. Mthode danalyse comparative. Enqute sur trois villes suisses*. CRESSON, 1991.
- [12] B. Odland and S. Auinger, "Reflections on the sonic commons," in *Leonardo Music Journal*, 19, 2009, p. 64.
- [13] Y.-F. Tuan, *Space and Place: The Perspective of Experience*. Minneapolis, London: University of Minnesota Press, 2001.
- [14] J. Cage, *An Autobiographical Statement*. Southwest Review, 1991.
- [15] Vesa. Norilo, "Kronos: A declarative metaprogramming language for digital signal processing," *Computer Music Journal*, vol. 39, no. 4, pp. 30–48, 2015.
- [16] —, "Kronos as a Visual Development Tool for Mobile Applications," in *Proceedings of the International Computer Music Conference*, Ljubljana, 2012, pp. 144–147.
- [17] H. Westerkamp, "Soundwalking," in *Sound Heritage, Volume III Number 4*, Victoria B.C., 1974 Revised 2001.
- [18] G. Debord, *Introduction to a Critique of Urban Geography*. Les Lvres Nues, 1955.
- [19] J.-P. Thibaud, "Installing an atmosphere," in *Architecture and Atmosphere*, ed. Philip Tidwell Tapio Wirkkala-Rut Bryk Foundation, 2014, 2014.
- [20] T. Ingold, "Against soundscape," in *Autumn leaves*, A. Carlyle ed., *Double Entendre*, 2007, pp. 10–13.
- [21] S. Feld, "Acoustemology," in *David Novak & Matt Saka-keeney, eds, Keywords in Sound*. Duke University Press, 2015.
- [22] C. Lattner and V. Adve, "LLVM: A compilation framework for lifelong program analysis & transformation," *International Symposium on Code Generation and Optimization 2004 CGO 2004*, vol. 57, no. c, pp. 75–86, 2004.

# A Framework for Multi- $f_0$ Modeling in SATB Choir Recordings

**Helena Cuesta**

Music Technology Group  
Universitat Pompeu Fabra  
helena.cuesta@upf.edu

**Emilia Gómez**

Universitat Pompeu Fabra and  
Joint Research Centre (EC)  
emilia.gomez@upf.edu

**Pritish Chandna**

Music Technology Group  
Universitat Pompeu Fabra  
pritish.chandna@upf.edu

## ABSTRACT

Fundamental frequency ( $f_0$ ) modeling is an important but relatively unexplored aspect of choir singing. Performance evaluation as well as auditory analysis of singing, whether individually or in a choir, often depend on extracting  $f_0$  contours for the singing voice. However, due to the large number of singers, singing at a similar frequency range, extracting the exact individual pitch contours from choir recordings is a challenging task. In this paper, we address this task and develop a methodology for modeling pitch contours of SATB choir recordings. A typical SATB choir consists of four parts, each covering a distinct range of pitches and often with multiple singers each. We first evaluate some state-of-the-art multi- $f_0$  estimation systems for the particular case of choirs with a single singer per part, and observe that the pitch of individual singers can be estimated to a relatively high degree of accuracy. We observe, however, that the scenario of multiple singers for each choir part (i.e. unison singing) is far more challenging. In this work we propose a methodology based on combining a multi- $f_0$  estimation methodology based on deep learning followed by a set of traditional DSP techniques to model  $f_0$  and its dispersion instead of a single  $f_0$  trajectory for each choir part. We present and discuss our observations and test our framework with different singer configurations.

## 1. INTRODUCTION

Singing in a SATB (Soprano, Alto, Tenor, Bass) choir is a long standing and well enjoyed practice, with many choirs following this format across different languages and cultures. Performances are based on scores, which provide linguistic, timing and pitch information for the singers in the choir to follow. Professional choirs practice for years to ensure that their performance is *in tune* with a reference pitch; however, due to the mechanism of voice production and expressive characteristics, the pitch of the individual voices in the choir often deviates from the theoretical pitch as indicated in the score. As a consequence, analysis and evaluation of a choir performance depends on the combination of pitches produced by individual singers in the choir. Through history, conductors, teachers, and

critics have relied on their own interpretation of pitch and harmony, while listening and/or evaluating a choir. In recent years, a few automatic analysis and evaluation systems have been proposed [1, 2] to provide an informed analysis of choirs in terms of intonation. In general, these systems require the extraction of accurate pitch contours for individual vocal tracks, which has hitherto been a road-block for analysis, as multi- $f_0$  extraction systems are not able to provide sufficient pitch precision and accuracy to drive analysis systems from full mixed choir recordings. This can primarily be pinned down to the fact that in a choral recording, multiple singers with similar timbres are singing in harmony, and even the same notes within each choir section, leading to overlapping harmonics, which are difficult to isolate. While several multi- $f_0$  estimation systems have been designed for music with easily distinguishable sources, e.g. music with vocals, guitar, bass and drums, very few research has been carried out in the domain of vocal ensembles, be it because of the lack of annotated datasets or because modeling several people singing very close frequencies, i.e. in unison, is very challenging in terms of  $f_0$  resolution.

In this work we address the computational modeling of pitch in choir recordings. In order to do that, we first evaluate how a set of multi- $f_0$  estimation algorithms perform with vocal quartets and try to identify their main limitations. Then, we use the evaluation results to select the best-performing algorithm and use it to extract a first approximation of the  $f_0$  of each choir section. In the second step we use a set of traditional DSP techniques to increase the pitch resolution around the estimated  $f_0$ s and model  $f_0$  dispersion. The main focus of this adaptation is not to obtain an accurate  $f_0$  estimate for each voice inside each choir section, but to model the distribution of  $f_0$  of a choir section singing in unison, measured through the dispersion of pitch values across each part.

The rest of the paper is organized as follows: Section 2 provides a brief overview of the current state-of-the-art for multi- $f_0$  extraction. Section 3 describes the limitations of current systems to characterize  $f_0$  in unison performances. Then, in Section 4 we define the evaluation metrics commonly used in the field, followed by Section 4.1 presenting the dataset used in this study. Section 5 discusses the initial evaluation of state-of-the-art methodologies on our particular material. Following this, Section 6 presents a novel approach to model unison recordings by combining a multi- $f_0$  estimation algorithm with traditional DSP techniques. Section 7 presents and discusses the results and

Copyright: © 2019 Helena Cuesta et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

limitations of the proposed system, and finally in Section 8 we provide some conclusions on the method and comments on future research that we intend to carry out.

## 2. STATE OF THE ART

Multi- $f_0$  estimation involves the detection of multiple concurrent  $f_0$  from an audio recording [3] and it is a core step of the task of automatic music transcription (ATM): converting an acoustic musical signal into some form of musical notation [4]. We briefly summarize a set of multi- $f_0$  estimation methods that can be applied to vocal music, although they try to estimate  $f_0$  values of individual singers, while we address the modeling of  $f_0$  of unison singing.

Duan et al. [5] presented an approach to multi- $f_0$  estimation using maximum-likelihood, where they model two different regions of the input power spectrum: the peak region, comprising the set of frequencies that are within a distance  $d$  of the peak frequencies, and the non-peak region, which is the complement of the peak region. The input signal is normalized and the power spectrum is computed frame-wise. A set of spectral peaks are extracted using a peak detection algorithm, and then several  $f_0$  candidates are computed in the range of one semitone around each peak. For each time frame, the  $f_0$  of each source is estimated by maximizing the probability of having harmonics that explain the observed peaks and minimizing the probability of having harmonics in the region where no peaks were observed. This is accomplished optimizing the parameters of a likelihood function that combines the peak region likelihood and the non-peak region likelihood, treated as independent sets. They use monophonic and polyphonic training data to learn the model parameters. Their system also estimates polyphony, i.e. how many sources there are in the mix, which define the number of  $f_0$  the model should estimate at each frame. Finally, a post-processing step using information for neighbouring frames is implemented to make the pitch predictions more stable. This process of refining the  $f_0$  estimates, however, removes duplicate estimates, which a problem in the case of several sources producing the same  $f_0$ , i.e. unison singing. The system parameters were learned using training data consisting of mixes of individual monophonic note recordings from 16 instruments including flute, saxophone, oboe, violin, bass, and violin among others. Then, they evaluated the algorithm on 4-part Bach chorales performed by a quartet: violin, clarinet, tenor saxophone and bassoon. These details about the data they used to train and evaluate the system are very relevant for our research, since given the lack of vocal data in the training and evaluation stages, we expect the system to perform worse in choral music.

Another relevant system for multiple  $f_0$  estimation is the one developed by Klapuri [6], which estimates each  $f_0$  in the mixture iteratively: at every step, the system detects the most predominant  $f_0$  and its corresponding harmonics are then subtracted from the spectrum. In this case, the input signal is passed through a bank of linear band-pass filters that resemble the inner ear behaviour in terms of frequency selectivity. Then, the output signal at each band is processed in a nonlinear manner to approximate

the firing activity in the auditory nerve. After this pre-processing steps, a frequency-domain signal representation is obtained by combining the band spectra, which is then used to extract a pitch salience function to emphasize the fundamental frequencies present in the signal. From this representation, multiple  $f_0$  values are estimated iteratively: at each step, a  $f_0$  value is estimated and its harmonic partials are removed from the spectrum. This step is repeated until a  $f_0$  value is estimated for each source. In [6] he also implements a polyphony estimator, which determines how many  $f_0$  values need to be extracted and therefore the number of iterations. The system was evaluated with a collection of mixtures of individual sounds (some of them from the same source as in [5]). The authors does not explicitly mention any vocals in the dataset, and therefore we assume the method is not optimized for our particular material.

Schramm and Benetos [7] presented a method specifically designed for multi- $f_0$  estimation in a *cappella* vocal ensembles. They use a two-step approach: first, a system based on probabilistic latent component analysis (PLCA) employs a fixed dictionary of spectral templates to extract the first frequency estimates; as a second step, a binary random forest classifier is used to refine the  $f_0$  estimates based on the overtones properties. Spectral templates are extracted from recordings of multiple singers singing pure vowels in English. These recordings belong to the RWC dataset [8]. This method uses the normalized variable-Q transform (VQT) as input, which is then factorized using the expectation-maximization algorithm to estimate the parameters of the model. As opposed to the previously presented methods, this one is focused on vocal ensembles and it is trained and evaluated with such data, i.e. vocal quartets, one singer per part. They use a  $f_0$  resolution of 20 cents, which is enough for transcription purposes but not to deal with frequencies as close to each other as in unisons.

Another method for multi- $f_0$  estimation designed for the case of multiple singers is the one presented by Su et al. [9]. The authors claim that data is crucial to develop and evaluate such systems, and yet there is not a labeled multi- $f_0$  dataset for choir, which is one of the most common type of music through the ages and cultures. Their work has two separate parts: first, they present a novel annotated dataset of choir and symphonic music; then, they build an unsupervised approach to multi- $f_0$  estimation using advanced time-frequency (TF) analysis techniques such as the concentration of time and frequency method. According to their paper, these techniques help improving the stabilization of pitch, which is interpreted in three dimensions: frequency, periodicity, and harmonicity.

Recent advancements in deep learning based systems have led novel deep learning based multi- $f_0$  extraction systems, designed to be agnostic to the exact source of the pitched instruments in the mix. *DeepSalience* [10] is one of the most recent systems for multi-instrument pop/rock songs and mixtures. The model leverages harmonic information provided by a HCQT transform, comprising 6 constant-Q transforms (CQT), with a convolutional neural network (CNN) to extract pitch salience from an input audio mixture. The network is fully convolutional with 5 convolu-

tional layers, it uses batch normalization and rectified linear units (ReLU) at each output. The final layer of the network uses logistic activation, mapping each bin of the output to the range  $[0,1]$ , representing pitch salience. It is trained using cross-entropy minimization. This pitch salience essentially predicts the probabilities of the underlying pitches being present in the input signal with a resolution of 20 cents. Then, using this salience intermediate representation, they use a threshold to estimate multiple frequencies at each frame.

These methods are capable of extracting multiple  $f_0$  from a great variety of audio signals, including music and speech. Most of them are designed for polyphonic signals where each melody is produced by a single source: one instrument or singer. However, the subject of our study are choirs, which involve unison ensemble singing, i.e. performances where several people sing the same notes. Unison recordings are challenging for multi- $f_0$  estimation because of the possible imprecision in the pitch produced by multiple singers or musicians [9]. Since we focus on the analysis and synthesis of choral singing, it is crucial to take this aspect into account to build models that consider these pitch imprecision.

### 3. PROBLEM DEFINITION AND APPROACH

The characterization of pitch distribution in unison and choir singing has not been widely studied. Most of the research in this topic is authored by Sundberg [11] and Ternström, who published a review on choir acoustics [12] and carried out several experiments to study  $f_0$  dispersion in unison singing [13]. The authors define  $f_0$  dispersion as the small deviations in  $f_0$  between singers that produce the same notes in a unison performance. This magnitude is directly related to the *degree of unison*, which Sundberg defines as the agreement between all the voices sources. In a later work, Jers and Ternström [14] measured the dispersion between singers and found it to range between 25 and 30 cents.

In multi- $f_0$  estimation systems, we usually focus on the extraction of a single pitch per source, and state-of-the-art algorithms would then provide a  $f_0$  value for each choir section. However, several singers produce slightly different values in each of the voices of a choir. Then, the question of which is the correct value to be estimated arises: most multi- $f_0$  estimation algorithms do not have enough resolution to discern the individual pitches, which leads to a potentially imprecise estimation. This suggests that unison performances need to be treated in a different way. Ternström [13] claims that while solo singing has tones with well-defined properties, i.e. pitch, loudness, timbre, unison ensemble singing has tones with statistical distributions of these properties, and we need to consider those when modeling them.

In a recent study, Cuesta and al. [1] created the Choral Singing Dataset (see Section 4.1) to analyze  $f_0$  dispersion in unison ensemble singing by modeling the distribution of fundamental frequencies. Using individual tracks for each singer, they extracted  $f_0$  curves and computed the mean  $f_0$  as the perceived pitch and the standard deviation of the

distribution as the  $f_0$  dispersion. In Figure 1 we display an example of the  $f_0$  trajectories of four sopranos, where we observe that there are slight  $f_0$  differences between them. This study found dispersion values ranging from 20 to 30 cents on average, depending on the choir section and the song, which agrees to previous literature [13]. However, this type of analysis requires an individual audio track for each singer in the mixture, and this data is difficult to obtain given that choirs are not recorded using this set up. This particular limitation leads us to explore in the present study ways of analyzing choir recordings directly from the singer mixture, which involves dealing with four different melodies (SATB), each of them involving a unison.

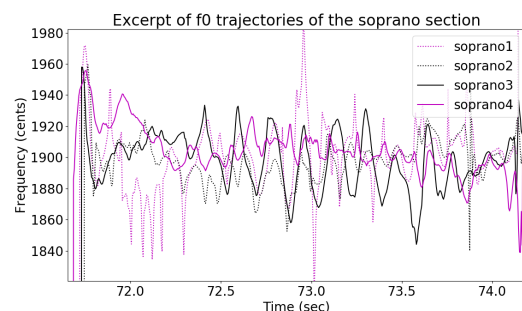


Figure 1:  $F_0$  curves of four sopranos singing the same note. We see how the curves oscillate and differ from each other.

In this study, we propose a methodology for pitch content analysis on unison ensemble singing that has two main stages:

1. **Multi- $f_0$  estimation.** In the first stage, we perform multi- $f_0$  estimation in the audio mixture in order to roughly estimate the pitches of the four voices of the choir. For this part, we evaluate the performance of a set of existing multi- $f_0$  estimation systems in the context of vocal quartets, where we have precise  $f_0$  ground truth information, to select the one with a better performance.
2.  **$F_0$ -dispersion modeling** In the second stage, we refine the frequency analysis around those pitches to further characterize  $f_0$  dispersion in each of the unison voices. In order to do so, we consider a DSP-based approach and adapt a method with higher frequency resolution to model each melodic source as a distribution of  $f_0$ s instead of a single value.

### 4. EVALUATION METHODOLOGY

As mentioned in previous sections, we evaluate the performance of three state-of-the-art algorithms for multi- $f_0$  estimation in vocal quartets, e.g. SATB with a single singer per section, in order to investigate which method is more suitable for this music material. We consider the methods proposed by Klapuri (KL) [6], Schramm et al. (SCH) [7] and Bittner et al. (DS) [10], all of them publicly available and representative of the state of the art in the area.



#### 4.1 Dataset

There are very few datasets of choral music which are annotated in terms of  $f_0$ . In our experiments, we take advantage of the Choral Singing Dataset<sup>1</sup> further described in [1]. This dataset was recorded in a professional studio and contains individual tracks for each of the 16 singers of a SATB choir, i.e. 4 singers per choir section. Although each section was recorded separately, synchronization between all audio tracks was achieved using a piano reference and a video of the conductor that singers followed during the recording.

This dataset comprises three different choral pieces: *Locus Iste*, written by Anton Bruckner, *Niño Dios d'Amor Herido*, written by Francisco Guerrero, and *El Rossinyol*, a Catalan popular song; all of them were written for 4-part mixed choir. This dataset is more suitable for this study than the one presented in [9]: having the individual tracks of each singer allows us to create *artificial* mixes between voices, e.g. vocal duets or quartets, small choir, large choir...etc. Using different combinations of all 16 singers, we created 256 SATB quartets for each piece, which represent all possible combinations of singers taking into account the voice type restriction, i.e. we need one singer per voice. These vocal quartets are used to evaluate the performance of the three algorithms.

#### 4.2 Multi- $f_0$ evaluation metrics

In multi- $f_0$  estimation systems, there are multiple  $f_0$  values per frame  $n$ . Following the terminology used by Bittner [15], we define the ground truth value(s) in frame  $n$  as  $f[n]$  and the estimation as  $\hat{f}[n]$ , which denote the pitches of all active sources in that frame.

For a given frame  $n$  we denote as true positives,  $TP[n]$ , the number of correctly transcribed pitches, and as false positives,  $FP[n]$ , the number of pitches present in the estimation,  $\hat{f}[n]$ , which are not present in the ground truth,  $f[n]$ . Similarly, the false negatives value,  $FN[n]$ , measures the number of pitches present in the ground truth which are not present in the estimation. Based on these, we define the following set of metrics: *accuracy*, *precision* and *recall*, and a set of errors: the substitution error ( $E_{sub}$ ), miss error ( $E_{miss}$ ), and false alarm error ( $E_{fa}$ ). Finally, total error,  $E_{tot}$  is reported as the combination of  $E_{sub}$ ,  $E_{miss}$  and  $E_{fa}$ .

All the presented evaluation metrics also have their associated *chroma* versions, which considers an estimated  $f_0$  to be correct if it is one octave apart from the corresponding target pitch. For more details about these metrics we refer the reader to [15].

### 5. MULTI- $F_0$ ESTIMATION RESULTS

All the SATB quartets of the dataset were evaluated in terms of multi- $f_0$  estimation: by means of the individual tracks, we extracted  $f_0$  curves for every singer using the spectral-amplitude autocorrelation (SAC) method [16] and

we then combined them to create the multiple  $f_0$  ground truth at each frame.

A summary of the results is displayed in Figure 2, where we present the accuracy, recall and precision averaged for each of the algorithms in the three songs of the dataset. We observe that *DeepSalience* (DS) outperforms Klapuri (KL) and Schramm (SCH). It is also interesting to point out that the difference between these metrics and their *chroma* versions is very small, thus suggesting that the three algorithms are fairly robust in terms of octave errors. We also observe that the algorithm by Schramm et al. has a higher variability with respect to the other ones, suggesting that its performance is highly dependent on the input signal. Also, it is important to mention that while KL and DS predict multi- $f_0$  values from a long audio file, i.e. a full choral recording, we splitted our audio material in shorter clips (each of them 10 seconds long) to evaluate SCH method: the PLCA algorithm employed in this method is computationally very expensive and we could not obtain results using the full recordings.

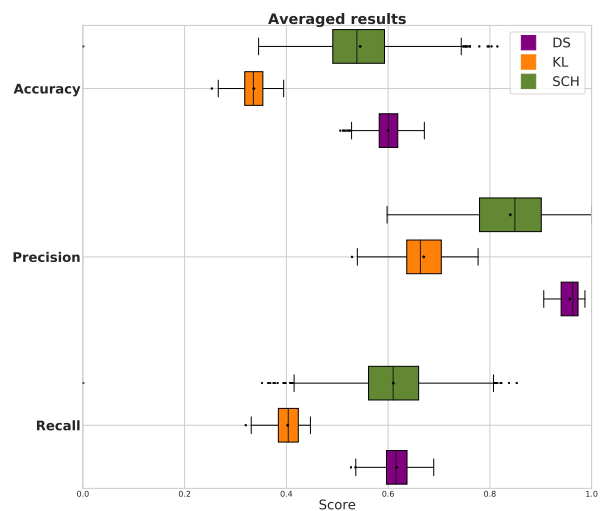


Figure 2: Accuracy, precision and recall for each of the three algorithms (DS, DU, SCH) averaged over all the dataset, i.e. all the SATB quartets.

In terms of error analysis, Table 1 provides the average errors for each of the algorithms. These results suggest that extracting multiple frequencies from a vocal ensemble is a very challenging task, since the total error is almost 40% in the best performing method. A part from this, we can extract a few more insights: all algorithms have a very low false alarm error, which means that they almost never report an  $f_0$  when there is not one in the ground truth; in addition, *DeepSalience* does a good job regarding the substitution error, which means that it rarely reports a wrong  $f_0$ . However, the miss error is pretty high, especially in Klapuri's algorithm, which means that there are a lot of  $f_0$  that are not extracted. In the case of Schramm's method, though, the miss error is lower, suggesting that their voice assignment step improves the performance of the multi- $f_0$

<sup>1</sup> Choral Singing Dataset: <https://zenodo.org/record/1319597>

	DS	SCH	KL
Substitution error	2.3%	10%	12%
Miss error	35%	28%	48%
False alarm error	0.4%	1.5%	8%
Total error	38%	40%	67%

Table 1: Summary of error metrics in multi- $f_0$  estimation .

estimation. We could also relate these differences to the fact that SCH is the only method designed for and trained with singing voice data. However, given the length limitation of SCH and based on the overall results, we select *DeepSalience* as the method to be used in the first stage of our model.

## 6. $F_0$ DISPERSION MODELING

The first step of our method presented above uses *DeepSalience* to extract multiple pitch estimations at each frame of the audio input. In the ideal case, at this stage we would obtain one  $f_0$  value for each choir section; however, as discussed in Section 5, although *DeepSalience* is the best-performing algorithm from the evaluated set, there are still some errors in the output.

In the second stage of our method we consider traditional DSP techniques to increase the frequency resolution of our model. We compute the spectrogram of the input audio signal using a Hanning window of 4096 points zero-padded to twice its length, resulting in an FFT size of 8192. An excerpt of this spectrogram is displayed in Figure 4 with magnitude in dB and the frequency axis in cents, where we observe that  $f_0$  values for each choir section are well-separated.

With this time-frequency representation, we then locate each of the estimated fundamental frequencies (*DeepSalience* output), which will ideally match one of the spectral peaks. Even though we use a large FFT size, since we want to obtain a high pitch resolution, we interpolate the peaks and recompute the peak locations as the maximum value of each interpolated peak. This process is illustrated in Figure 3, where the top and bottom plots correspond to a vocal quartet and full choir spectrum, respectively. The dashed black line represents the original spectrum, while the red solid lines and the green asterisk correspond to the interpolated peaks. We observe that the peaks in the full choir case (bottom) have less energy and are a bit more noisy than the vocal quartet ones (see third and sixth peaks for example).

Once we have this information, we compute the bandwidth of each peak as a measure of the dispersion of the  $f_0$  distribution in the unison case. Remember that our aim is to characterize the distribution of  $f_0$  for each choir section rather than obtaining a single  $f_0$  value. For each choir section, we find and interpolate the peak in the spectrum and consider the peak frequency as the mean frequency of the distribution and its bandwidth as its dispersion. The bandwidth is expressed in cents (computed with a refer-

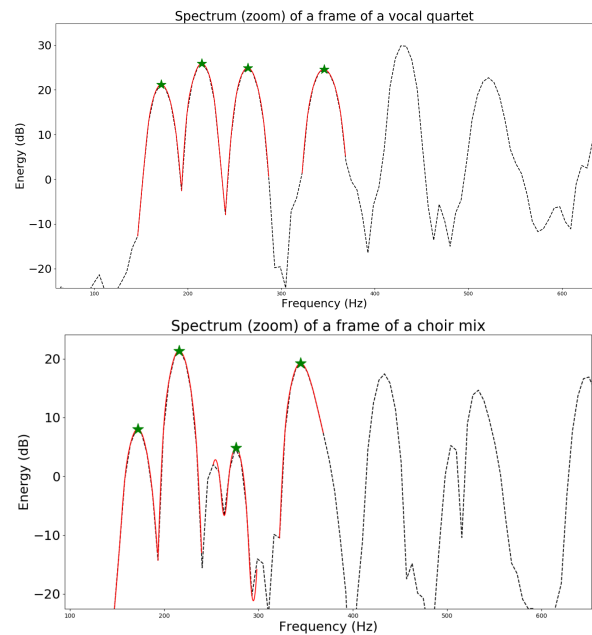


Figure 3: Example frames of the spectrum with the interpolated peaks corresponding to the estimated  $f_0$ s (green asterisk). The top plot corresponds to a 4-singers audio input (quartet), while the bottom plot corresponds to the 16-singer audio input (full choir).

ence frequency of 220 Hz) and computed as follows:

$$f_{0\text{dispersion}} = b_2 - b_1 \quad (1)$$

where  $b_2$  and  $b_1$  are the frequency bins around the spectral peak where the amplitude of the spectrum decays 3 dB. Note that in this first approach we do not take into account the window type and size used in the spectral analysis, although they influence the peak bandwidth. In further studies, we plan to study and document the effect of these two analysis parameters in the dispersion computation.

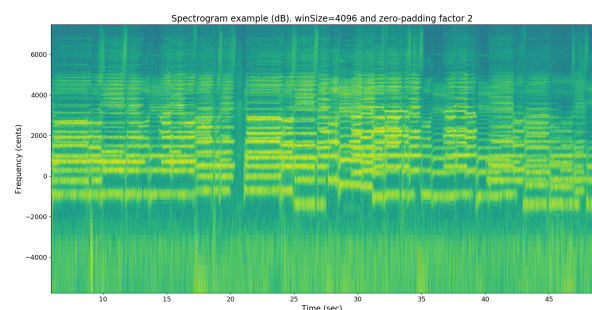


Figure 4: Spectrogram (zoomed) of the piece Locus Iste computed using a Hanning window and  $N = 4096$  zero-padded to twice its length, resulting in an FFT size of 8192.

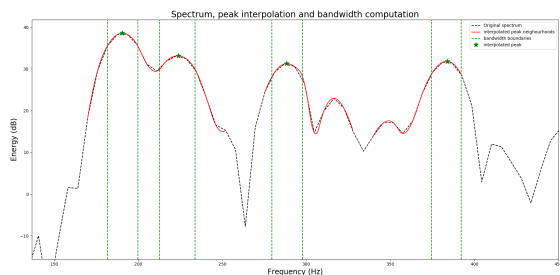


Figure 5: Example frame of the spectrum with the interpolated peaks and the boundaries for the bandwidth computation.

## 7. RESULTS

In this Section we present the results of the dispersion analysis averaged by piece and by choir section. Since the first step of the presented framework outputs noisy multi- $f_0$  estimations, the results of the second step displayed here are obtained using the ground truth pitch values to locate the peaks. This allows us to perform a better evaluation of the dispersion computation method, since the errors that come from the multi- $f_0$  estimation are not present here. In the real application, the ground truth pitches would be replaced by the output of the selected multi- $f_0$  estimation algorithm. Figure 5 shows another example frame of the analysis, with the peak interpolation and also vertical lines showing the bandwidth delimitation.

Table 2 provides the  $f_0$  dispersion results averaged by choir section and piece. *BTAS* stand for *bass*, *tenor*, *alto*, *soprano*, and *Q* and *CM* are short for *quartet* and *choir mix*, meaning that the dispersion values belong to a 4-singers and 16-singers setting, respectively. We would expect the dispersion values to be larger in the 16 singers case, because having several voices producing similar frequencies might generate wider spectral peaks. The differences on average are not very strong; however, we conducted an independent-samples t-test to compare the performances of vocal quartets with the full choir and found that the differences were significant. For example, there was a significant difference in the bass section of the quartet singing *El Rossinyol* ( $M=181, SD=39$ ) and the bass section in the choir mix ( $M=191, SD=79$ ),  $\alpha = 0.05, p < 0.001$ . This tendency applies to the whole dataset, although the effect size is small (around 0.2 on average) and therefore we might need a deeper analysis to find out if the differences are not only statistically significant but also relevant in terms of pitch content.

These results can not be directly compared to any ground truth, since previous studies modeled pitch dispersion in different ways, i.e. standard deviation of the distribution [1] or bandwidth of the partial tones [13]. Instead, we compare the tendency of the results with the ones obtained by the authors in a previous study where individual pitch tracks, and thus accurate individual  $f_0$  estimations, were used [1]. In Table 3 we display the results from the men-

	Locus Iste		El Rossinyol		Niño Dios	
	Q	CM	Q	CM	Q	CH
<b>B</b>	231 (57)	248 (130)	181 (39)	191 (70)	227 58	257 175
<b>T</b>	136 (30)	140 (38)	132 (26)	136 (30)	143 (31)	149 (40)
<b>A</b>	100 (22)	104 (25)	98 (19)	103 (23)	105 (22)	110 (28)
<b>S</b>	76 (23)	79 (27)	75 (16)	80 (20)	78 (20)	82 (25)

Table 2: Average  $f_0$  dispersion results computed as the bandwidth of the peaks in the whitened spectrum. Dispersion values are in **cents**. *Q* refers to a SATB quartet with one singer per section and *CM* refers to a SATB choir mix with 4 singers per section. Values in italics are standard deviations also in cents.

Soprano	Alto	Tenor	Bass
20.16	22.66	22.22	26.02
Locus Iste		El Rossinyol	Niño Dios
19.32		27.65	21.33

Table 3: Averaged results of pitch dispersion from [1]. All values represent dispersion in cents, computed as the standard deviation of the pitch distribution.

tioned work, where the dispersion is averaged by choir section (all three pieces together) and by piece (all four choir sections together). These results suggest that the dispersion (in cents) is higher in the bass section of a choir, which is also true for our results. Following basses, the presented results have tenors, followed by altos and finally sopranos, with the lowest average dispersion. In Table 3 we have altos and tenors with very similar average values, and sopranos also obtained the lowest dispersion values. Therefore, although the dispersion magnitude can not be compared, the trend is very similar, suggesting that the analysis is consistent.

After an analysis of the results, we observe that this framework has a few limitations, including that the results highly depend on the performance of the multi- $f_0$  estimation algorithm employed in the first stage. In this paper we evaluated three algorithms which are considered state-of-the-art and *DeepSalience* was selected according to the evaluation we carried out, but we hypothesize that using a system specially designed for singing voice, and even for choral singing, e.g. [7] and [9], might improve the performance of the whole method: if the  $f_0$  estimates at each frame are precise, then the peaks would be modelled correctly (as happened in the dispersion evaluation), yielding more accurate  $f_0$  dispersion values. However, these methods were dismissed from the final approach because of the length limitation [7] and because it is not publicly available [9].

The proposed framework models the  $f_0$  distribution of a unison using two values: the  $f_0$ , extracted in the first stage and refined in the second stage, and the  $f_0$  dispersion. We

believe these values are a good descriptor for unison performances, but a more complex model incorporating temporal evolution analysis could also be considered and used to estimate the quality of a choir or in choral synthesis applications.

## 8. CONCLUSIONS

In this paper we presented a framework for the  $f_0$  modeling in choral singing recordings that uses a two-stage methodology: first, a deep learning based multi- $f_0$  estimation system is employed to obtain one pitch value for each choir section; second, we locate these frequencies in a whitened version of the spectrum of the input signal with a higher pitch resolution. This process allows us to model each unison as a distribution of  $f_0$ , characterized by two values: the mean  $f_0$  and the  $f_0$  dispersion. The preliminary results we obtained do not show strong differences in the dispersion between a large and a small number of singers, but more data might reveal different patterns. Since we evaluated this framework as a case study, more experiments will be done in the near future, and a deeper analysis of the relationship between the time-frequency representation and the results will be carried out to make the system more effective.

## Acknowledgments

This work is partially supported by the European Commission under the TROMPA project (H2020 770376) and by Spanish Ministry of Economy and Competitiveness under the CASAS project (TIN2015-70816-R). First author is supported by FI Predoctoral Grant from AGAUR (Generalitat de Catalunya). We thank Rodrigo Schramm and Andrew McLeod for the support running their code with our audio material.

## 9. REFERENCES

- [1] H. Cuesta, E. Gómez, A. Martorell, and F. Loáiciga, "Analysis of Intonation in Unison Choir Singing," in *Proc. of the 15th International Conference on Music Perception and Cognition (ICMPC)*, Graz (Austria), 2018.
- [2] J. Dai and S. Dixon, "Analysis of Interactive Intonation in Unaccompanied Satb Ensembles," in *18th International Society for Music Information Retrieval Conference*, 2017.
- [3] A. McLeod, R. Schramm, M. Steedman, and E. Benetos, "Automatic Transcription of Polyphonic Vocal Music," *Applied Sciences*, vol. 7, no. 12, p. 1285, 2017. [Online]. Available: <http://www.mdpi.com/2076-3417/7/12/1285>
- [4] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, "Automatic Music Transcription: Challenges and Future Directions," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 407–434, 2013.
- [5] Z. Duan, B. Pardo, and C. Zhang, "Multiple Fundamental Frequency Estimation by Modeling Spectral Peaks and Non-peak Regions," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 2121–2133, 2010.
- [6] A. Klapuri, "Multiple Fundamental Frequency Estimation by Summing Harmonic Amplitudes," in *Proc. of the International Society for Music Information Retrieval Conference*, Victoria, Canada, 2006, pp. 216–221.
- [7] R. Schramm, E. Benetos *et al.*, "Automatic Transcription of A Cappella Recordings from Multiple Singers." Audio Engineering Society, 2017.
- [8] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Music Genre Database and Musical Instrument Sound Database," 2004.
- [9] L. Su, T.-y. Chuang, and Y.-h. Yang, "Exploiting Frequency, Periodicity and Harmonicity Using Advanced Time-Frequency Concentration Techniques for Multi-pitch Estimation of Choir and Symphony," in *Proc. International Society for Music Information Retrieval Conference*, 2016.
- [10] R. M. Bittner, B. McFee, J. Salamon, P. Li, and J. P. Bello, "Deep Saliency Representations for F0 Estimation in Polyphonic Music," in *Proc. of the International Society for Music Information Retrieval Conference*, New York, 2017, pp. 63–70.
- [11] J. Sundberg, *The Science of the Singing Voice*. Northern Illinois University Press, 1987.
- [12] S. Ternström, "Choir acoustics: an overview of scientific research published to date," *Speech, Music and Hearing Quarterly Progress and Status Report*, vol. 43, no. April, pp. 001–008, 2002.
- [13] —, "Perceptual evaluations of voice scatter in unison choir sounds," *STL-Quarterly Progress and Status Report*, vol. 32, pp. 041–049, 1991.
- [14] H. Jers and S. Ternström, "Intonation analysis of a multi-channel choir recording," *TMHQPSR Speech, Music and Hearing: Quarterly Progress and Status Report*, vol. 47, no. 1, pp. 1–6, 2005.
- [15] R. Bittner, "Data-driven fundamental frequency estimation," Ph.D. dissertation, New York University, 2018.
- [16] E. Gómez and J. Bonada, "Towards Computer-Assisted Flamenco Transcription: An Experimental Comparison of Automatic Transcription Algorithms as Applied to A Cappella Singing," *Computer Music Journal*, vol. 37, no. 2, pp. 73–90, 2013.



# Representations of self-coupled modal oscillators with time-varying frequency

Tamara Smyth, Jennifer S. Hsu  
Department of Music, UC San Diego  
trsmth@ucsd.edu

## ABSTRACT

In this work we examine a simple mass-spring system in which the natural frequency is modulated by its own oscillations, a self-coupling that creates a feedback system in which the output signal “loops back” with an applied coefficient to modulate the frequency. This system is first represented as a mass-spring system, then as an extension of well-known frequency modulation synthesis (FM) coined “loopback FM”, and finally, as a closed-form representation that has a form similar to the transfer function of a “stretched” allpass filter with time-varying delay, but with the fundamental difference that it is used here as a time-domain signal, the real part of which is the sounding waveform. This final representation allows for integration of instantaneous frequency in the FM representation and ultimately a mapping from its parameters to those of loopback FM. In addition to predicting the sounding frequency (pitch glides) of loopback FM for a given carrier frequency and time-varying loopback coefficient, or equivalently of the self-coupled oscillator for a given natural frequency and coupling coefficient, the closed form representation is seen to be a more accurate representation of the system as it does not introduce a unit-sample delay in the feedback loop, nor is it as numerically sensitive to sampling rate.

## 1. INTRODUCTION

It is well known that introducing nonlinearities into a linear system may contribute computational complexity making it prohibitive for real-time use [1,2]. If, however, the aim is to apply the dynamic sound effects of nonlinear coupling to a synthesized sound, prioritizing real-time parametric control over acoustic accuracy, there are representations in the literary canon of parametric abstract synthesis techniques that can be explored. In spirit and purpose similar to [3], this work explores the relationship between a physically self-coupled oscillator to the well-known abstract synthesis technique, frequency modulation (FM).

As shown in Section 2, a simple physical model of a two-degree-of-freedom (2-DOF) mass-spring system that exhibits modal coupling behaviour, can be “abstracted” and represented as a simplified self-coupled oscillator, one in which the mass influences its own oscillation in a feedback system. Discretization of this oscillator’s displacement, yielding a second-order bandpass filter, is problem-

atic when the frequency, and thus the filter coefficient dependent on frequency, is made time varying. The system is shown in Section 4 to be strongly related to FM, and to resolve issues of filter instability, the self-coupled oscillator is presented in terms of a variant of FM so called “loopback FM” [4] to distinguish it from the related, but distinct, “feedback FM” [5]. Finally, the closed-form representation of the loopback FM oscillator is given, offering improved numerical accuracy, eliminating the need for a unit-delay in the feedback loop and, perhaps most advantageous for musical applications, revealing the nonlinear oscillator’s sounding frequency. The musical application of this work is explored in a related paper by the same authors, whereby a modal synthesis model of percussion instruments is implemented using loopback FM oscillators, allowing for a linear model to be enhanced by the rich and dynamic sounds caused by nonlinear modal coupling that are characteristic of these instruments [6].

## 2. A COUPLED OSCILLATOR

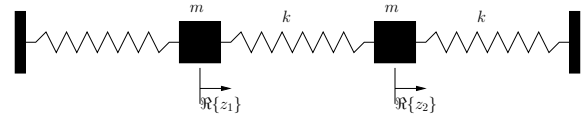


Figure 1. A two-degree-of-freedom oscillator having two masses and three springs of equal value, where the displacement of each mass is given by  $\Re\{z_{1,2}\}$ .

The equations of motion for a two-degree-of-freedom mass-spring oscillator with mass  $m$  and spring constant  $k$  are

$$\begin{aligned} m\ddot{z}_1 + kz_1 + k(z_1 - z_2) &= 0 \\ m\ddot{z}_2 + kz_2 + k(z_2 - z_1) &= 0, \end{aligned} \quad (1)$$

with displacement of each mass being given by  $\Re\{z_{1,2}(t)\}$ , and  $z_{1,2}(t)$  having assumed solutions

$$z = Ae^{j\omega t}, \dot{z} = j\omega Ae^{j\omega t}, \ddot{z} = -\omega^2 Ae^{j\omega t}, \quad (2)$$

and where (1) has 2 natural modes of oscillations: one where  $A_1 = A_2$ :

$$-\omega^2 A_1 + \omega_0^2 A_1 + \omega_0^2 (A_1 - A_2) = 0 \quad \text{and} \quad \omega = \omega_0 \quad (3)$$

and the other where  $A_1 = -A_2$ :

$$-\omega^2 A_1 + \omega_0^2 A_1 + \omega_0^2 (A_1 - A_2) = 0 \quad \text{and} \quad \omega = \sqrt{3}\omega_0, \quad (4)$$

Copyright: © 2019 Tamara Smyth, Jennifer S. Hsu et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

where  $\omega_0 = \sqrt{k/m}$ . In addition to the 2 natural modes,  $z_{1,2}$  may exhibit coupled behaviour (e.g. given a specific set of initial conditions) in which one mass influences the oscillations of the other. To explore coupled behaviour, we begin with a generalized parametric expression in which frequency is modulated by the oscillations of the system

$$\omega = \omega_0 + d_1 \Re\{z_1\} + d_2 \Im\{z_1\} + d_3 \Re\{z_2\} + d_4 \Im\{z_2\}, \quad (5)$$

where coupling coefficients  $d_{1,2,3,4}$  specify the amount of frequency deviation contributed by the oscillations of each mass (considering both real and imaginary parts of complex  $z_{1,2}$ ). In this work, a special simplified case of (5) is explored where  $d_{2,3,4} = 0$  and  $\omega = \omega_0 + d_1 \Re\{z_1\}$ , and the oscillator is merely coupled to itself, creating a system that will be later referred to in Section 4 (and was previously coined in [4]) as “loopback FM”.

Since frequency  $\omega$  is now made time varying, the relationship between instantaneous frequency  $\omega_i(t)$  and instantaneous phase  $\theta_i(t)$ ,

$$\omega_i(t) = \frac{d}{dt} \theta_i(t) \quad \text{and} \quad \theta_i(t) = \int_0^t \omega_i(t) dt, \quad (6)$$

must be considered before defining assumed solution  $z_1$ :

$$z_1(t) = \exp\left(j \int_0^t (\omega_0 + d_1 \Re\{z_1(t)\}) dt\right) \quad (7)$$

a system for which sounding frequency is not as easily predicted as in (2). Furthermore, if  $d_1$  is made time varying, the sounding frequency will change over time, resulting in a pitch glide—a known characteristic of nonlinearly coupled systems—having a trajectory dependent on the nature of the function  $d_1(t)$ .

## 2.1 Assumed solution

Given the more general assumed solution  $z_1(t) = e^{j\theta(t)}$ , its first and second derivatives with respect to time are given by

$$\dot{z}_1(t) = j\dot{\theta}(t)z_1(t), \quad (8)$$

$$\begin{aligned} \ddot{z}_1(t) &= j\ddot{\theta}(t)z_1(t) + j\dot{\theta}(t)\dot{z}_1(t) \\ &= (j\ddot{\theta}(t) - \dot{\theta}(t)^2)z_1(t), \end{aligned} \quad (9)$$

with the angle and its derivatives given by

$$\theta(t) = \int_0^t \omega_0 + d_1 \Re\{z_1(t)\} dt, \quad (10)$$

$$\dot{\theta}(t) = \omega_0 + d_1 \Re\{z_1(t)\}, \quad (11)$$

$$\begin{aligned} \ddot{\theta}(t) &= d_1 \Re\{\dot{z}_1(t)\} \\ &= d_1 \Re\{j\dot{\theta}(t)z_1(t)\} \\ &= d_1 \Re\{j\dot{\theta}(t)\Re\{z_1(t)\} - \dot{\theta}(t)\Im\{z_1(t)\}\} \\ &= -d_1 \dot{\theta}(t)\Im\{z_1(t)\}. \end{aligned} \quad (12)$$

The equation of motion adapted from (1) for a single mass-spring oscillator

$$m\ddot{z}_1(t) + kz_1(t) = 0 \quad (13)$$

in which the spring constant is modulated such that

$$\sqrt{k(t)/m} = \omega_0 + d_1 z_1(t) \quad (14)$$

$$k(t) = m(\omega_0 + d_1 z_1(t))^2 \quad (15)$$

may be represented as

$$\ddot{z}_1(t) + (\omega_0 + d_1 z_1(t))^2 z_1(t) = 0, \quad (16)$$

which, having additional terms  $2\omega_0 d_1 z_1^2$  and  $d_1^2 z_1^3$ , is now nonlinear in  $z_1$ . To verify that the interpretation of  $k(t)$  given in (14-15) satisfies the equation of motion for a self-coupled (feedback) oscillator, equation (9) is first substituted for  $\ddot{z}_1(t)$  in (16),

$$j\ddot{\theta}(t) - \dot{\theta}(t)^2 = -(\omega_0 + d_1 z_1(t))^2, \quad (17)$$

and (12) substituted for  $\ddot{\theta}(t)$  in (17) to yield

$$\begin{aligned} -jd_1 \dot{\theta}(t)\Im\{z_1(t)\} - \dot{\theta}(t)^2 &= -(\omega_0 + d_1 z_1(t))^2 \\ \dot{\theta}(t) (jd_1 \Im\{z_1(t)\} + \dot{\theta}(t)) &= (\omega_0 + d_1 z_1(t))^2 \end{aligned} \quad (18)$$

where, by (11), the LHS parenthetical expression in (18) is

$$jd_1 \Im\{z_1(t)\} + \omega_0 + d_1 \Re\{z_1(t)\} = \omega_0 + d_1 z_1(t) \quad (19)$$

to finally yield

$$\begin{aligned} \dot{\theta}(t)(\omega_0 + d_1 z_1(t)) &= (\omega_0 + d_1 z_1(t))^2 \\ \dot{\theta}(t) &= \Re\{\omega_0 + d_1 z_1(t)\}, \end{aligned} \quad (20)$$

showing an instantaneous frequency equal to (11) when it is assumed to be real, thus further showing  $\sqrt{k(t)/m} = \omega_0 + d_1 z_1(t)$  satisfies the equation of motion.

## 2.2 Implementation of mass-spring oscillator

One known solution for the discretization of the mass-spring oscillator is using the trapezoidal rule for numerical integration (or bilinear transform). A version of (13) that is driven by force function  $F_k(t) = F(t)/k$ :

$$\ddot{z}_1(t) + \omega_0^2 z_1(t) = F_k(t), \quad (21)$$

has  $s$ -transform

$$s^2 Z_1(s) + \omega_0^2 Z_1(s) = F_k(s), \quad (22)$$

and transfer function

$$H(s) = \frac{Z_1(s)}{F_k(s)} = \frac{1}{s^2 + \omega_0^2}, \quad (23)$$

which, when taking the  $z$ -transform by substituting  $s$  with  $\frac{1-z^{-1}}{c \frac{1+z^{-1}}{2}}$ , where  $c = 2/T$  without prewarping, yields

$$H(z) = \left( \frac{1}{c^2 + \omega_0^2} \right) \frac{1 + 2z^{-1} + z^{-2}}{1 - 2 \frac{c^2 - \omega_0^2}{c^2 + \omega_0^2} z^{-1} + z^{-2}}. \quad (24)$$

The numerator and denominator polynomials of (24) may be expressed in polar form as,

$$B(z) = (1 + z^{-1})^2 \quad (25)$$

$$A(z) = (1 - az^{-1})(1 - a^* z^{-1}), \quad (26)$$

where roots of  $A(z)$  (poles of  $H(z)$ ) are the complex conjugate pair having sum

$$2 \frac{c^2 - \omega_0^2}{c^2 + \omega_0^2} = \frac{(c - j\omega_0)^2 + (c + j\omega_0)^2}{(c + j\omega_0)(c - j\omega_0)} = \underbrace{\frac{c - j\omega_0}{c + j\omega_0}}_a + \underbrace{\frac{c + j\omega_0}{c - j\omega_0}}_{a^*}. \quad (27)$$

The gain of filter  $H(z)$  is given by

$$G(\omega) = |H(\omega)| = \frac{1}{c^2 + \omega_0^2} \frac{|B(\omega)|}{|A(\omega)|}, \quad (28)$$

where

$$\begin{aligned} |B(\omega)| &= \left| \left( e^{-j\omega T/2} \left( e^{j\omega T/2} + e^{-j\omega T/2} \right) \right)^2 \right| \\ &= \left| e^{-j\omega T} (2 + e^{j\omega T} + e^{-j\omega T}) \right| \\ &= 2(1 + \cos(\omega T)), \end{aligned} \quad (29)$$

$$\begin{aligned} |A(\omega)| &= \left| e^{-j\omega T/2} \left( e^{j\omega T/2} - a e^{-j\omega T/2} \right) \right| \times \\ &\quad \left| e^{-j\omega T/2} \left( e^{j\omega T/2} - a^* e^{-j\omega T/2} \right) \right| \\ &= \left| e^{-j\omega T} (-a - a^* + e^{j\omega T} + e^{-j\omega T}) \right| \\ &= \left| -2 \left( \frac{c^2 - \omega_0^2}{c^2 + \omega_0^2} - \cos(\omega T) \right) \right|, \end{aligned} \quad (30)$$

and (28) reduces to

$$G(\omega) = \frac{1 + \cos(\omega T)}{|c^2 - \omega_0^2 - (c^2 + \omega_0^2) \cos(\omega T)|}. \quad (31)$$

Transfer function (24) is a linear-time-invariant second-order bandpass filter having, as shown by (31), a spectral peak in the magnitude at  $\omega_0$  and taking the inverse transform of (24) yields an undamped sinusoidal oscillator that closely matches (21). Though (24) is well behaved for static  $\omega_0$ , it has problems when made time varying. It could be made tuneable by introducing a loss as in [7], but this would have consequences when placed in a feedback system where the loss would accrue.

### 2.3 Discrete-Time Complex Oscillator

A point in the complex plane  $z_s(0) = A e^{j\phi_0}$  can be made to rotate with angle  $\omega_i T$  via a complex multiply

$$z_s(1) = e^{j\omega_i T} A e^{j\phi_0}, \quad (32)$$

or equivalently, as shown in [4], using a power preserving rotational matrix. If  $\omega_i$  is static (indicated here by subscript  $s$ ), regular rotations every time sample  $n = 0, 1, \dots, N - 1$  produces an oscillator given by the complex sinusoid:

$$z_s(n) = (e^{j\omega_i T})^n A e^{j\phi_0} = A e^{j(\omega_i n T + \phi_0)}, \quad (33)$$

having instantaneous phase  $\omega_i n T + \phi_0$ , initial phase  $\phi_0$ , and instantaneous angular frequency  $\omega_i$ . If however,  $\omega_i$  is made time varying, the representation in (33) no longer applies and the relationship between frequency and phase given in (6) must be considered before defining the oscillator. For example, if the oscillator frequency changes linearly from  $\omega_1$  to  $\omega_2$  over  $T_d$  seconds, the instantaneous phase would be given by

$$\int_0^t \left( \frac{\omega_2 - \omega_1}{T_d} t + \omega_1 \right) dt = \frac{\omega_2 - \omega_1}{2T_d} t^2 + \omega_1 t + C, \quad (34)$$

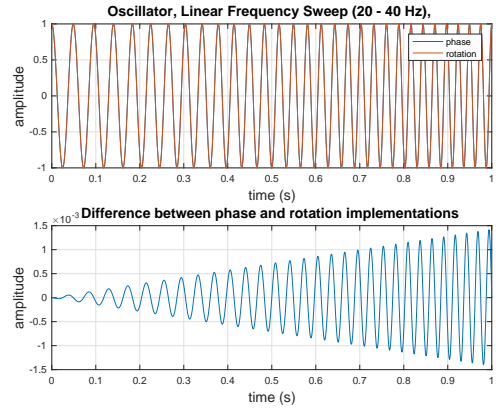


Figure 2. Though implementing an oscillator via a rotation of its previous sample (akin to a numerical integration of frequency) produces a similar result to implementing the phase analytically (top), there is a difference, and the error observed (bottom) can compound.

yielding the discrete-time oscillator (by substitution  $t \rightarrow nT$ ) notably different from (33):

$$z_l(n) = \exp \left[ j \left( \left( \frac{\omega_2 - \omega_1}{2T_d} nT + \omega_1 \right) nT + \phi_0 \right) \right], \quad (35)$$

where initial phase  $\phi_0$  is set to the constant of integration  $C$ . It is possible to implement this oscillator via sample-by-sample rotations of angle  $\omega_i(n)T$  though *not* of an initial complex value as in (33), but rather of its current state:

$$\begin{aligned} e^{j\omega_i(n)T} z_l(n-1) &= e^{j\omega_i(n)T} e^{j\omega_i(n-1)T} \dots e^{j\omega_i(0)T} A e^{j\phi_0} \\ &= \exp \left[ j \sum_{m=0}^n \omega_i(m)T \right] A e^{j\phi_0}. \end{aligned} \quad (36)$$

The summation in (36) can also be viewed as a numerical integration of the instantaneous frequency, which, as shown in Figure 2, comes at a cost of numerical error when compared to using the instantaneous phase (34) directly. Though in many cases this error is negligible (and inaudible), it is the reason why, as discussed in the following section, it is often preferable to implement frequency modulation (FM) as phase modulation (PM), especially in cases where networks of multiple carriers can cause such error to compound.

### 3. FM/PM REPRESENTATION

In the well-known synthesis technique first introduced by Chowning [8], the frequency/phase of an oscillator may be made to change sinusoidally, introducing sidebands about a carrier frequency and changing the sound's spectrum in a way that's dependent on the amplitude, phase, and frequency of the modulating sinusoid. In frequency modulation (FM) synthesis, a carrier oscillator has a center frequency  $\omega_c$  that is modulated by a sinusoid having amplitude  $d$  and frequency  $\omega_m$ , yielding instantaneous frequency

$$\omega_i(t) = \omega_c + d \cos(\omega_m t), \quad (37)$$

where  $d$  determines the oscillator's peak frequency deviation from  $\omega_c$ . Notably, (37) has a form very similar to (5)

with  $d_{2,3,4} = 0$ , and this will be developed in the next section. The corresponding instantaneous phase is obtained by integrating (37) according to (6), yielding

$$\theta_i(t) = \int_0^t \omega_i(t) dt = \omega_c t + \frac{d}{\omega_m} \sin(\omega_m t) + \phi_c, \quad (38)$$

showing that FM may be equivalently expressed as phase modulation (PM), where it is the initial phase term that is sinusoidally time varying,

$$\phi(t) = I \sin(\omega_m t) + \phi_c, \quad (39)$$

with amplitude

$$I = \frac{d}{\omega_m}, \quad (40)$$

a value known as the *index of modulation* because of how it influences the magnitude of sidebands at  $f_c \pm k f_m$  in the resulting spectrum. FM synthesis is frequently implemented as PM, preferred because of improved numerical properties (such as those illustrated in Figure 2) and accuracy less dependent on sampling rate, with the real signal being given by

$$x_c(t) = \cos(\omega_c t + I \sin(\omega_m t)), \quad (41)$$

or, as the real part of the complex exponential sinusoids,

$$x_c(t) = \Re\{z_c(t)\} = \Re\left\{e^{j(\omega_c t + \Im\{z_m(t)\})}\right\}, \quad (42)$$

where

$$\Im\{z_m(t)\} = I \sin(\omega_m t). \quad (43)$$

Using the complex form has the power-preserving advantage discussed above and in [4], and allows FM to be represented as a sample-by-sample rotation of its current state, shown by beginning with (42) at time sample  $n - 1$ :

$$z_c(n - 1) = e^{j(\omega_c(n - 1)T + \Im\{z_m(n - 1)\})}, \quad (44)$$

then adding and subtracting  $\Im\{z_m(n)\}$  to its argument

$$\begin{aligned} \angle z_c(n - 1) &= j(\omega_c n T + \Im\{z_m(n)\} - \omega_c T \\ &\quad - \Im\{z_m(n)\} + \Im\{z_m(n - 1)\}) \\ &= \angle z_c(n) \\ &\quad - j(\omega_c T + \Im\{z_m(n) - z_m(n - 1)\}) \end{aligned} \quad (45)$$

so that  $z_c(n - 1)$  may be represented first as a multiplication by  $z_c(n)$

$$z_c(n - 1) = z_c(n) e^{-j(\omega_c T + \Im\{z_m(n) - z_m(n - 1)\})}, \quad (46)$$

and then finally in its causal form

$$z_c(n) = e^{j(\omega_c T + \Im\{z_m(n) - z_m(n - 1)\})} z_c(n - 1). \quad (47)$$

Notably, this result is equivalent to taking the derivative of the phase with respect to continuous-time  $t$  to produce instantaneous frequency,

$$\omega_i(t) = \frac{d}{dt} (\omega_c t + \Im\{z_m(t)\}) = \omega_c + \frac{d}{dt} \Im\{z_m(t)\}, \quad (48)$$

then using a finite different approximation to obtain its discrete-time form

$$\omega_i(n) = \omega_c + \frac{\Im\{z_m(n)\} - \Im\{z_m(n - 1)\}}{T}, \quad (49)$$

which, when normalized by the sampling period  $T$  yields the angle of rotation in (47).

#### 4. SELF COUPLING AND LOOPBACK FM

Applying the above to the self-coupled oscillator in (5) where  $d_{2,3,4} = 0$  and the instantaneous frequency is  $\omega_i(t) = \omega_0 + d_1 \Re\{z_1(t)\}$ , it is evident from (44 - 47) that this system may be expressed as a sample-by-sample rotation of its current state, where the carrier oscillator is “looped back” to serve as the modulator of its frequency, with added unit sample delay necessary for implementation:

$$z_c(n) = e^{j(\omega_c + B\omega_c \Re\{z_c(n - 1)\})T} z_c(n - 1), \quad (50)$$

and modulation amplitude  $B\omega_c = d$  determines the peak frequency deviation from  $\omega_c$ , while the loopback coefficient  $B$  functions as the index of modulation according to (40).

A more accurate representation of (50) is expected of one in which a delay of one sample is not required and which does not essentially implement a numerical approximation of the system’s instantaneous phase, as shown by (47 - 49). Integrating the instantaneous frequency  $\omega_c + B\omega_c \Re\{z_c(t)\}$  with respect to continuous-time  $t$  yields an alternate representation of the system in which the corresponding instantaneous phase is given by

$$\begin{aligned} \theta_i(t) &= \int_0^t \omega_c + B\omega_c \Re\{z_c(t)\} dt \\ &= \omega_c t + B\omega_c \Re\left\{\int_0^t z_c(t) dt\right\}. \end{aligned} \quad (51)$$

Though the integral term in (51) may be implemented via numerical integration to yield the discrete-time representation of instantaneous phase:

$$\theta_i(n) = \omega_c n T + B\omega_c T \Re\left\{\sum_{k=0}^{n-1} z_c(k)\right\}, \quad (52)$$

this solution does not improve upon—and in fact is exactly equal to—equation (50) when incorporated into the phase modulation representation  $e^{j\theta_i(n)}$ . Furthermore, it does not provide greater understanding of the system’s behaviour, and in particular, reveal at what frequency it will sound. It is preferable, therefore, to solve (51) analytically.

##### 4.1 Analytic Solution to $\theta_i(t)$ for static $B$

Figure 3 plots the real part of  $z_c(n)$  given in (50), showing a periodic signal having a period of  $M$  samples and a sounding frequency of  $f_0 = f_s/M$  Hz, a signal that can also be described by the real part of

$$z_0(n) = \frac{b_0 + e^{j\omega_0 n T}}{1 + b_0 e^{j\omega_0 n T}}, \quad (53)$$

where  $\omega_0 = 2\pi f_s/M$ . Equation (53) is similar in form to the transfer function of the “stretched” allpass filter used in



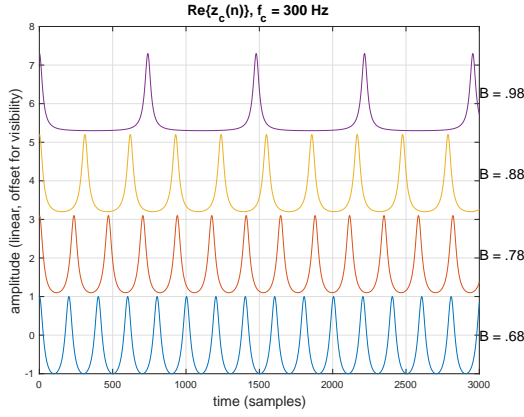


Figure 3. The real part of  $z_c(n)$  given in (50) is plotted (with offset) for 4 linearly spaced values of  $B$ , showing a *nonlinear* relationship to resulting period in samples  $M$  (and sounding frequency  $f_0 = f_s/M$  Hz,  $f_s = 44.1$  kHz).

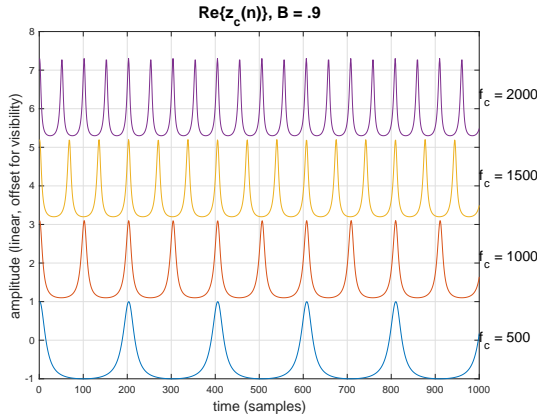


Figure 4. The real part of  $z_c(n)$  given in (50) is plotted with 4 linearly spaced values of  $f_c = \omega_c/(2\pi)$  showing a *linear* relationship with resulting period in samples  $M$  (and sounding frequency  $f_0 = f_s/M$  Hz,  $f_s = 44.1$  kHz).

[9], though here it is used as a time-domain signal that is a function of time sample  $n$ —a complex oscillator of which we ultimately take the real part to produce the sounding waveform. It is also interesting to observe a similarity between the waveforms in Figures 3-5 and those produced by feedback amplitude modulation (FBAM) [10] for input  $\cos(\omega_0 nT)$ , as well as the related time series in [11]. Though the pulse shape and offset are indeed different, their similarity does suggest further study of their relationship would be worthwhile.

Though Figure 6 shows  $z_0(n)$  and  $z_c(n)$  diverging for increased values of  $f_c = \omega_c/(2\pi)$  and  $B$  (not shown), this is improved with increased sampling rate (reducing numerical error as well as the effect of the unit-sample delay in (50)), providing confidence that  $z_0(n)$  is actually the preferred and more accurate solution to the self-coupled oscillator. With this assumption, the integral of  $z_c(t)$  with respect to continuous-time  $t$  in (51) may now be expressed analytically by the integral of continuous-time  $z_0(t)$  (obtained by substitution  $nT \rightarrow t$  in  $z_0(n)$  given in (53)) which, as shown in Appendix A for static  $\omega_0$  and  $b_0$ , is

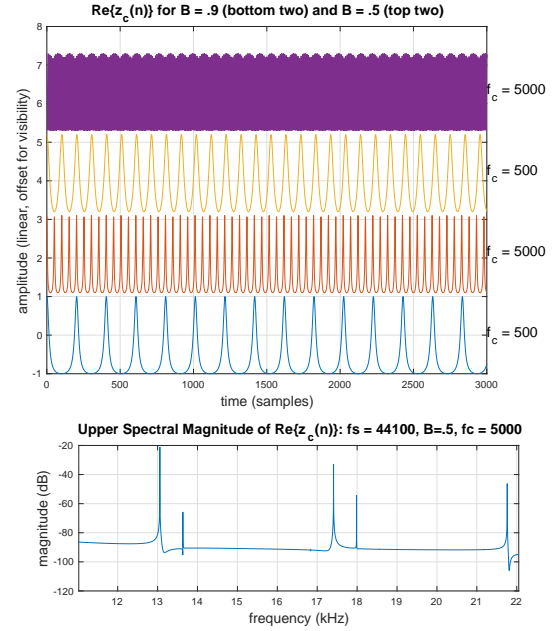


Figure 5. Higher values of  $f_c$  with lower values of  $B$  can produce signals that show low-frequency amplitude modulation (beating) most prominent here when  $B = .5$  and  $f_c = 5000$  Hz (top). The beat period ( $\approx 80$  samples) and beat frequency ( $\approx f_s/80 = 551$ ,  $f_s = 44.1$  kHz), is also visible as the frequency difference (bottom) between higher frequency components closer to the Nyquist limit  $f_s/2$ . This strongly suggests aliasing, and artifacts disappear when  $f_s$  is increased.

given by

$$\int_0^t z_0(t) dt = b_0 t + \frac{1 - b_0^2}{j\omega_0 b_0} \log(1 + b_0 e^{j\omega_0 t}). \quad (54)$$

Representing the term inside the logarithm in polar form

$$1 + b_0 e^{j\omega_0 t} = A(t) e^{j\phi(t)}, \quad (55)$$

where

$$A(t) = \sqrt{1 + 2b_0 \cos(\omega_0 t) + b_0^2} \quad (56)$$

and

$$\phi(t) = \tan^{-1} \left( \frac{b_0 \sin(\omega_0 t)}{1 + b_0 \cos(\omega_0 t)} \right), \quad (57)$$

the real part of (54) may be expressed as

$$\begin{aligned} \Re \left\{ \int_0^t z_0(t) dt \right\} &= b_0 t + \\ &\Re \left\{ \frac{1 - b_0^2}{j\omega_0 b_0} (\log(A(t)) + j\phi(t)) \right\} \\ &= b_0 t + \frac{1 - b_0^2}{\omega_0 b_0} \phi(t), \end{aligned} \quad (58)$$

and the final expression for phase  $\theta_i(t)$  in (51) becomes

$$\begin{aligned} \theta_i(t) &= \omega_c t + B\omega_c \Re \left\{ \int_0^t z_c(t) dt \right\} \\ &= \omega_c t (1 + Bb_0) + B\omega_c \frac{1 - b_0^2}{\omega_0 b_0} \phi(t). \end{aligned} \quad (59)$$

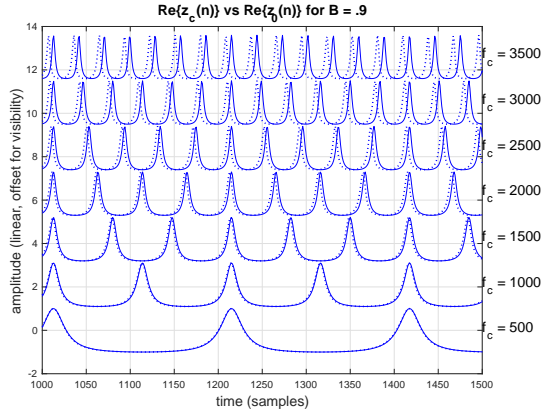


Figure 6.  $\Re\{z_c(n)\}$  (solid) and  $\Re\{z_0(n)\}$  (broken) show increasingly less agreement over time for higher  $f_c$  and higher  $B$  (not shown) as they drift out of phase.

for yet unknown values  $b_0$  and  $\omega_0$ , solved in terms of loopback FM parameters  $f_c$  and  $B$  in the following section.

#### 4.2 Mapping $b_0$ and $\omega_0$ to Loopback FM Parameters

Expressions for parameters  $b_0$  and  $\omega_0$  may be obtained by first setting the angle of  $z_0(t)$ , well known but derived in Appendix B as

$$\angle z_0(t) = \omega_0 t - 2 \tan^{-1} \left( \frac{b_0 \sin(\omega_0 t)}{1 + b_0 \cos(\omega_0 t)} \right), \quad (60)$$

equal to the instantaneous phase of the loopback FM representation given in (59):

$$\omega_c t (1 + B b_0) + B \omega_c \frac{1 - b_0^2}{\omega_0 b_0} \phi(t) = \omega_0 t - 2\phi(t), \quad (61)$$

where  $\phi(t)$  is given in (57). Setting linear terms on the left- and right-hand side (LHS and RHS) of (61) to be equal, yields one expressions for  $\omega_0$ :

$$\omega_0 = \omega_c (1 + B b_0), \quad (62)$$

while setting LHS and RHS oscillating terms to be equal yields a second expression for  $\omega_0$ :

$$\omega_0 = \frac{\omega_c B (1 - b_0^2)}{-2b_0}. \quad (63)$$

Setting (62) equal to (63) yields the quadratic equation

$$B b_0^2 + 2b_0 + B = 0, \quad (64)$$

where  $b_0$  is given in terms of loopback FM parameter  $B$ :

$$b_0 = \frac{\pm \sqrt{1 - B^2} - 1}{B}. \quad (65)$$

Finally, substituting (65) into (62) yields an expression for  $\omega_0$  as a function of loopback FM parameters  $B$  and  $\omega_c$ :

$$\omega_0 = \omega_c \left( 1 + B \frac{\pm \sqrt{1 - B^2} - 1}{B} \right) = \pm \omega_c \sqrt{1 - B^2}. \quad (66)$$

#### 4.3 Allowing for Time-Varying Sounding Frequency

The derivation in the previous section assumes a static loopback variable  $B$  which, by (66), also produces a static sounding frequency  $\omega_0$  and static  $b_0$ . To produce a change in sounding frequency over time,  $B$  must be made time varying and the expression for  $z_0(t)$  made more generalized:

$$z_0(t) = \frac{b_0 + e^{j\theta_0(t)}}{1 + b_0 e^{j\theta_0(t)}}, \quad (67)$$

where the argument of the exponential terms is the integral with respect to time of time-varying frequency  $\omega_0(t)$ :

$$\theta_0(t) = \int_0^t \omega_0(t) dt = \pm \int_0^t \omega_c \sqrt{1 - B(t)^2} dt. \quad (68)$$

and  $\omega_0(t)$  is adapted from (66) for time-varying  $B(t)$ . Clearly the expression resulting from (68) is dependent on the nature of function  $B(t)$ .

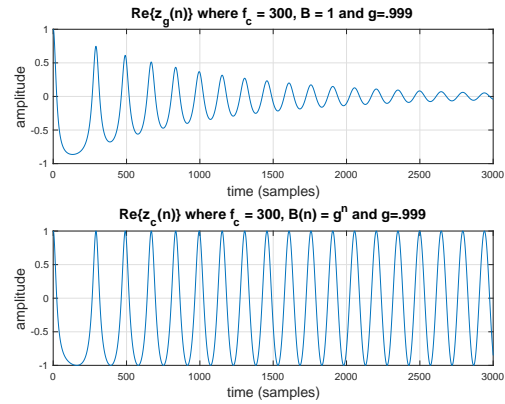


Figure 7. Waveform  $\Re\{z_g(n)\}$  (top) show that introducing a scalar multiple  $g$  to the loopback FM equation (69) introduces a change in both sounding frequency and amplitude. Waveform  $\Re\{z_c(n)\}$  (bottom) shows that setting the feedback coefficient to  $B(n) = g^n$  produces the same pitch change but without imposing an amplitude envelope.

Consider the case where the complex oscillator is multiplied by a scalar value  $g$  such that when it is looped back, its amplitude envelope decays exponentially:

$$z_g(n) = g e^{j(\omega_c + \omega_c \Re\{z_g(n-1)\}) T} z_g(n-1), \quad (69)$$

where here  $B = 1$ . As shown in Figure 7, and evident from (69), the system will have both an amplitude envelope and a time-varying frequency, the latter equal to that of the loopback FM oscillator (50) in which  $B$  is made time varying with exponential function

$$B(n) = g^n, \quad n = 0, 1, \dots, N-1. \quad (70)$$

Representing loopback FM with time-varying  $B(n)$  instead of (69) allows amplitude and (sounding) frequency envelopes to be divorced and independently described. If  $B(n)$  is exponentially time varying according to (70),  $\theta_0$  may be adapted from (68) and represented as a function of discrete-time sample  $n$ :

$$\theta_0(n) = \int_0^n \omega_0(n) T dn = \pm \int_0^n \omega_c T \sqrt{1 - g^{2n}} dn, \quad (71)$$

which, as shown in Appendix C, yields final expression

$$\theta_0(n) = \frac{\omega_c T}{\log(g)} \left( \sqrt{1 - g^{2n}} - \tanh^{-1}(\sqrt{1 - g^{2n}}) + C \right), \quad (72)$$

where  $C$  is an integration constant. Of course a different solution would result for  $\theta_0(n)$  if  $B(n)$  were made to change linearly with sample  $n$ :

$$B_l(n) = kn + l, \quad n = 0, 1, \dots, N, \quad (73)$$

yielding

$$\begin{aligned} \theta_0(n) &= \int_0^n \omega_c \sqrt{1 - B_l(n)^2} dn \\ &= \frac{\omega_c T}{2k} \left( B_l(n) \sqrt{1 - B_l(n)^2} + \sin^{-1} B_l(n) \right) + C. \end{aligned} \quad (74)$$

Figure 8 shows the spectrum of  $z_0(n)$  overlaid with a dark curve plotting time-varying fundamental frequency  $f_0 = f_c \sqrt{1 - B(n)}$  for  $B(n) = g^n$  (top) and  $B(n) = kn + l$  (bottom). The close fit between curve and lowest spectral harmonic shows that the fundamental frequency of loopback FM can be accurately predicted for both static and time-varying  $B$  and, in the latter case, use of (72) and (74) in expression for  $z_0(n)$  in (67) is valid.

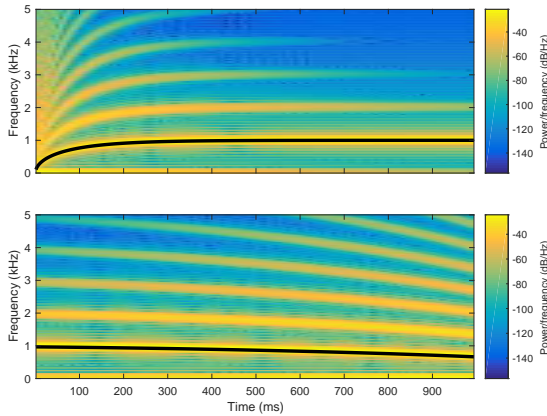


Figure 8. The spectrum of  $z_0(n)$  with time-varying argument  $\theta_0(n)$  overlaid with fundamental frequency  $f_c \sqrt{1 - B(n)}$  (dark curve) for  $B(n) = g^n$  (top) and  $B(n) = kn + l$  (bottom), validating use of  $\theta_0(n)$  in  $z_0(n)$  for known functions  $B(n)$  and showing sounding frequency of loopback FM can be accurately predicted.

It is clear that when  $B$  changes, so does the expression for  $\theta_0$ , which might be seen as a limitation of this approach, except that it could be argued there are only a few ways in which one would expect  $B$  to change, and these can be expressed as functions with more subtle changes being accomplished via parameters settings. Furthermore, it is always possible to apply a numerical integration scheme if an analytical solution is not available.

Finally, it should also be noted that though it is possible to set a desired trajectory for sounding frequency  $\omega_0(t)$  in  $z_0(n)$ , there is no guarantee this will be mappable to loopback FM parameters and the oscillator given by (50).

## 5. CONCLUSIONS

This work explored possible representations of the non-linearly self-coupled oscillator, laying the groundwork for analysis and synthesis of systems with coupling in multiple modes. Beginning with the physical representation of the mass-spring system, an implementation of the oscillator using the bilinear transform is proposed, producing a biquadratic resonant filter that, without loss, is marginally stable and not well behaved when made time varying. Nevertheless, the assumed solution for equation of motion with frequency  $\omega_0 + d_1 \Re\{z_t(t)\}$ , which could serve as an implementation when made discrete, is shown to be valid.

The modulation of the oscillator's frequency is formulated with a variant of FM synthesis called loopback FM, whereby the carrier oscillator loops back to serve as a modulator of its own frequency. Because of the integral relationship between frequency and phase, an alternate more numerically accurate closed-form representation of the system is required to produce an analytical solution to the oscillator's phase, ultimately revealing it's sounding frequency. This closed-form representation of the loopback FM oscillator is presented first in its static case, yielding mappings between the parameters of the two representations, and then in its more general form to allow for time-varying sounding frequency.

## Acknowledgments

This work was strongly motivated by a conversation with Miller Puckette who had implemented a version of the system (5) in the real-time programming environment Pd. Inspired by the dynamics of the produced sound, the authors set out to describe it mathematically, with the hope the results could be used to enhance other synthesis models having similar feedback nonlinear characteristics. The authors are also sincerely grateful to the SMC reviewers whose in-depth comments and thorough reviews have greatly improved the presentation of this work.

## Appendix A

The integral of  $z_0(t)$  with respect to  $t$  may be represented as the sum of two integral terms:

$$\int_0^t z_0(t) dt = \int_0^t \frac{b_0}{1 + b_0 e^{j\omega_0 t}} dt + \int_0^t \frac{e^{j\omega_0 t}}{1 + b_0 e^{j\omega_0 t}} dt. \quad (75)$$

The first term of (75) may be represented by

$$\int_0^t \frac{b_0}{1 + b_0 e^{j\omega_0 t}} dt = b_0 \int_0^t \left( \frac{1 + b_0 e^{j\omega_0 t} - b_0 e^{j\omega_0 t}}{1 + b_0 e^{j\omega_0 t}} \right) dt, \quad (76)$$

which, when employing u-substitution where

$$u = 1 + b_0 e^{j\omega_0 t}, \quad \frac{du}{dt} = j\omega_0 b_0 e^{j\omega_0 t}, \quad dt = \frac{du}{j\omega_0 b_0 e^{j\omega_0 t}}, \quad (77)$$

may be further expressed as

$$\begin{aligned} \int_0^t \frac{b_0}{1 + b_0 e^{j\omega_0 t}} dt &= b_0 t - b_0 \int_0^t \frac{b_0 e^{j\omega_0 t}}{u} \frac{du}{j\omega_0 b_0 e^{j\omega_0 t}} \\ &= b_0 t - \frac{b_0}{j\omega_0} \int_0^u \frac{1}{u} du \\ &= b_0 t - \frac{b_0}{j\omega_0} \log(u). \end{aligned} \quad (78)$$

The second term of (75) is given by

$$\begin{aligned} \int_0^t \frac{e^{j\omega_0 t}}{1 + b_0 e^{j\omega_0 t}} dt &= \int_0^t \frac{e^{j\omega_0 t}}{u} \frac{du}{j\omega_0 b_0 e^{j\omega_0 t}} \\ &= \frac{1}{j\omega_0 b_0} \int_0^u \frac{1}{u} du \\ &= \frac{1}{j\omega_0 b_0} \log(u). \end{aligned} \quad (79)$$

Summing (78) and (79) and substituting values for  $u$  in (77) yields the final expression for the integral of  $z_0(t)$ :

$$\int_0^t z_0(t) dt = b_0 t + \frac{(1 - b_0^2) \log(1 + b_0 e^{j\omega_0 t})}{j\omega_0 b_0}. \quad (80)$$

## Appendix B

The angle of an expression have the form

$$H(\theta) = \frac{c + e^{j\theta}}{1 + ce^{j\theta}}, \quad (81)$$

is given by

$$\begin{aligned} \angle H(\theta) &= \angle(c + e^{j\theta}) - \angle(1 + ce^{j\theta}) \\ &= \angle(e^{j\theta}(1 + ce^{-j\theta})) - \angle(1 + ce^{j\theta}) \\ &= \angle e^{j\theta} + \angle(1 + ce^{-j\theta}) - \angle(1 + ce^{j\theta}) \\ &= \theta - 2 \tan^{-1} \left( \frac{c \sin(\theta)}{1 + c \cos(\theta)} \right). \end{aligned}$$

## Appendix C

Employing u-substitution where  $u = \sqrt{1 - g^{2n}}$  and

$$\frac{du}{dn} = -\frac{1}{2}(1 - g^{2n})^{-1/2} 2g^{2n} \log(g), \quad (82)$$

it follows that  $-g^{2n} = u^2 - 1$  and

$$dn = -\frac{\sqrt{1 - g^{2n}}}{g^{2n} \log(g)} du = \frac{u}{\log(g)(u^2 - 1)} du, \quad (83)$$

so that  $\theta_0(n)$  in (71) may be expressed as

$$\theta_0(n) = \pm \int_0^n \omega_c T u dn = \frac{\omega_c T}{\log(g)} \int_0^u \frac{u^2}{u^2 - 1} du. \quad (84)$$

The integral term in (84) may be solved as

$$\begin{aligned} \int_0^u \frac{u^2}{u^2 - 1} du &= \int_0^u \left( 1 - \frac{1}{(1 + u)(1 - u)} \right) du \\ &= \int_0^u 1 du - \int \left( \frac{(1 + u) + (1 - u)}{2(1 + u)(1 - u)} \right) du \\ &= u - \frac{1}{2} \int_0^u \left( \frac{1}{1 - u} + \frac{1}{1 + u} \right) du. \end{aligned}$$

Making substitutions

$$s = 1 - u, ds = -du \quad \text{and} \quad p = 1 + u, dp = du \quad (85)$$

yields

$$\begin{aligned} \int_0^u \frac{u^2}{u^2 - 1} du &= u + \frac{1}{2} \int_0^s \frac{1}{s} ds - \frac{1}{2} \int_0^p \frac{1}{p} dp \\ &= u - \frac{1}{2} (\log(p) - \log(s)) \\ &= u - \frac{1}{2} \log \left( \frac{1 + u}{1 - u} \right) + C \\ &= u - \tanh^{-1}(u) + C, \end{aligned}$$

where  $C$  is the integration constant, and the final expression for  $\theta_0(n)$  is

$$\theta_0(n) = \frac{\omega_c T}{\log(g)} \left( \sqrt{1 - g^{2n}} - \tanh^{-1} \left( \sqrt{1 - g^{2n}} \right) + C \right). \quad (86)$$

## A. REFERENCES

- [1] F. Avanzini and R. Marogna, "A modular physically based approach to the sound synthesis of membrane percussion instruments," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 891–902, May 2010.
- [2] S. Bilbao, "A family of conservative finite difference schemes for the dynamical von karman plate equations," *Numerical Methods for Partial Differential Equations*, vol. 24, no. 1, pp. 193–216, January 2008.
- [3] J. R. Pierce and S. A. V. Duyne, "A passive nonlinear digital filter design which facilitates physics based sound synthesis of highly nonlinear musical instruments," *Journal of the Acoustical Society of America*, vol. 101, no. 2, February 1997.
- [4] T. Smyth and J. Hsu, "On phase and pitch in loopback frequency modulation with a time-varying feedback coefficient," in *Proceedings of the 26<sup>th</sup> International Congress on Sound and Vibration*, Montreal, Canada, July 2019.
- [5] N. Tomisawa, "Tone production method for an electronic musical instrument," 1981, uS Patent 4,249,447. [Online]. Available: <http://www.google.com/patents/US4249447>
- [6] J. Hsu and T. Smyth, "Percussion synthesis using loopback frequency modulation oscillators," in *Proceedings of the 16<sup>th</sup> Sound and Music Computing Conference*, Málaga, Spain, May 2019.
- [7] J. O. Smith, "A constant-gain digital resonator tuned by a single coefficient," *Computer Music Journal*, vol. 6, no. 4, pp. 36–40, 1982.
- [8] J. Chowning, "The synthesis of complex audio spectra by means of frequency modulation," *Journal of the Audio Engineering Society*, vol. 21, no. 7, pp. 526–534, September 1973.
- [9] V. Välimäki, J. Parker, and J. Abel, "Parametric spring reverberation effect," *Journal of the Audio Engineering Society*, vol. 58, no. 7/8, pp. 547–562, 2010.
- [10] J. Kleimola, V. Lazzarini, V. Välimäki, and J. Timoney, "Feedback amplitude modulation synthesis," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, no. Article ID 434378, p. 18 pages, January 2011.
- [11] M. LeBrun, "Digital waveshaping synthesis," *Journal of the Audio Engineering Society*, vol. 27, no. 4, 1979.



# SONAGRAPH. A CARTOONIFIED SPECTRAL MODEL FOR MUSIC COMPOSITION

Andrea Valle

CIRMA/StudiUm - Università di Torino

andrea.valle@unito.it

## ABSTRACT

This paper presents SonaGraph, a framework and an application for a simplified but efficient harmonic spectrum analyzer suitable for assisted and algorithmic composition. The model is inspired by the analog Sonagraph and relies on a constant-Q bandpass filter bank. First, the historical Sonagraph is introduced, then, starting from it, a simplified (“cartoonified”) model is discussed. An implementation in SuperCollider is presented that includes various utilities (interactive GUIs, music notation generation, graphic export, data communication). A comparison of results in relation to other tools for assisted composition is presented. Finally, some musical examples are discussed, that make use of spectral data from SonaGraph to generate, retrieve and display music information.

## 1. INTRODUCTION

Access to spectral information is crucial for a large number of audio-related practices spread along the sound/music continuum, such as sound synthesis and processing [1, 2], audio restoration [3], composition for acoustic instruments (algorithmic/assisted composition [4, 5]), music information retrieval [6]. The great majority of available applications are based on Fourier transform, as implemented by the Fast Fourier Transform (FFT) algorithm. The efficiency (in computational terms) and the effectiveness (in terms of results) of the FFT are well known. While data gathered via FFT allow for reconstruction and manipulation of the input signal (audio level), they are not immediately suited for perceptual and music tasks, so that various post-processing operations are needed to extract music information [6]. In contrast to the audio level, this higher level, both perceptual and musical, might be called –following [7]– sound object level (see also [8]). Starting from FFT data, in order to pass from the first level to the second, a double conversion step has to be taken into account, that concerns both time and frequency [6]. With respect to FFT, time resolution is defined by the hop size (the step size in which the window is to be shifted across the signal), while the phenomenological one instead concentrates on a (not obvious) notion of sound event. The other step is required to convert frequencies (in Hertz) into

a pitch-based representation (a typical but not necessary example being 12-TET). This step is notoriously problematic because FFT samples frequency in a linear fashion, while the tonotopic representation in the ear, thus at the basis of musical practices, is logarithmic. This means that half of the information output from an FFT concerns, in perceptual terms, the highest octave, a quarter the second octave, and so on. In essence, high frequencies are over-represented and/or low frequencies are under-represented [2]. Spectral information is typically inspected visually, and many data visualization softwares and libraries are available to generate sonographic (i.e. spectrum over time) representations. Well-known applications targeted at the music domain are Sonic Visualiser [9], Acousmographie<sup>1</sup>, ianalyse5<sup>2</sup>: as they all rely on FFT, they all provide a sonographic visualization based on a linear distribution of frequencies. This issue is often partially solved by drawing logarithmically the frequencies, but since the starting information is linear with respect to frequency, the result is usually a sonogram that looks blurred in the low register while becoming very detailed in the highest one. To sum up, FFT spectral data are in principle too large and only partially fitting if the sound object level is at stake. While typically these issues are practically solved with satisfying results, still they leave room for different designs.

## 2. MAIN GOALS AND INTENDED USERS

SonaGraph is a spectral analyzer that, by means of a very basic design, aims at providing a spectral representation in form of a sketch, assuming that a sketch results in minimal but correct, clear and relevant information about a certain sound object. Because of its barebone structure, such an analyzer is efficient as it performs a large data reduction. While not fitted for audio manipulation (because of data loss), it allows to gather and easily manipulate musically relevant data from the musician’s perspective (i.e. at the sound object level). Its design is inspired by the so-called “cartoonification”, proposed in the audio domain in relation to the modelling of acoustic behaviour [10]. Cartoonification is a procedure that leads to a drastically simplified model that nevertheless remains consistent with some general principles. SonaGraph is intended as a tool bridging the audio level to the symbolic one by providing a simplified spectral data structure that can be easily understood by musicians without a particularly deep knowledge of the

Copyright: 2019 Andrea Valle. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> <http://inagrm.com/en/showcase/news/203/acousmographie>

<sup>2</sup> <https://logiciels.pierrecouprie.fr>

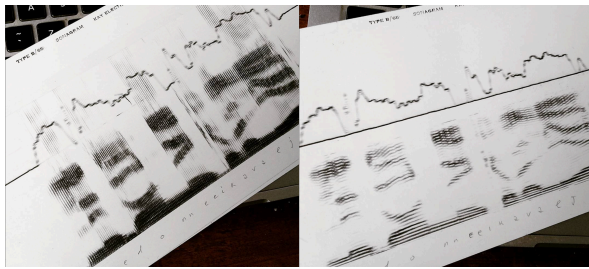


Figure 1. Sonagraph tracings: large (left) and narrow (right) bandwidth filter setting.

mathematics at the basis of more accurate methods and frameworks (e.g. [11]). As spectral information is immediately converted into pitch, standard music symbolic concepts and operation (e.g. chords and the relative typical manipulation) can be directly applied by the musician to spectral data. Moreover, thanks to the one-to-one correspondence between spectral data and music symbols, automated music notation can be generated easily while precisely representing the reduced spectrum. In the following, the general framework is introduced, then an application is presented. As musicians are the intended target, a comparison with BACH [12], a state-of-the-art tool used by composer in the same context (assisted composition including spectral analysis and automatic notation transcription) is provided. Finally, two musical applications are discussed.

### 3. THE ANALOG SONAGRAPH

The SonaGraph model is inspired by the Sonagraph, an analog device initially developed at Bell Labs in telecommunication field [13] and commercialized by Kay Electric from the '50s. The Sonagraph allowed to plot a representation of sound spectrum over time (a "sonagram"). Its hardware implementation was based on a bank of heterodyne filters performing a multistep spectrum scan (thus, it did not work in real time), connected to a stylus that burned progressively (i.e. frequency by frequency) a special paper foil [13, 14]. Time resolution depended on the (adjustable) rotation speed of the cylinder on which the paper was drawn. Features of the Sonagraph were the possibility of using two different bandwidths (large and narrow) and the ability to plot frequencies in a dual mode, that is, linearly and logarithmically. Figure 1 shows two examples of Sonagraph's tracings for the same signal respectively with large and narrow bandwidth setting for the filter bank, both plotted with linear frequency (each tracing also includes the signal's amplitude envelope on top).

By making sound structure extensively accessible for the first time, the Kay Sonagraph allowed fundamental advances in two domains. The first is acoustic phonetics, starting from the 1952 pioneering study of Jakobson, Fant and Halle [15], that, by means of sonagraphic exploration, led to the acoustic definition of phoneme (even if the very first use can be traced back to 1947 [16]). The second is bioacoustics, and in particular ornithology. Once available, the Sonagraph immediately became a fundamental instru-

ment for the study of bird singing as it provided visible and annotable forms to represent the extraordinary variety of ornithological phonations [14]. For Pieplow, this "golden age" of the Sonagraph extended for more than forty years, from 1951 to 1995<sup>3</sup>. A third application has been crucial for the fate of contemporary music, and thus it is particularly relevant here. In the domain of acoustic analysis of music instruments, Leipp made abundant and pioneering use of the Sonagraph to exemplify a vast set of acoustic phenomena and music instrument behaviours [17]: Leipp's approach was a fundamental pivot for the reflection that would lead during the '70s Grisey and his companions to the technical and aesthetic formulation of Spectralism [18]. The Sonagraph for the first time showed sound as a sort of virtual score: a form of notation in which notes are replaced by continuous graphic elements [19]. Bridging ornithology and music, Mâche's zoomusicological approach largely relied on sonagraphic traces [20].

Indeed, since more than twenty years the Sonagraph has been superseded by FFT-based softwares [14]. Interestingly, Leipp [17] strongly supported the perceptual appropriateness of the sonagraphic display even if all his examples were plotted linearly. In the ornithological field, on the contrary, Marshall insisted with equal emphasis on the perceptual requirement of a logarithmic frequency display, due to the tonotopic organization of the auditory system, a feature common to all vertebrates, including both birds and humans. For Marshall, linear frequency displaying was simply "absurd" [21].

### 4. A CARTOONIFIED MODEL

The Sonagraph provides a reference for a "cartoonified" model of spectrum analysis based on a filter bank. The idea of a filter bank that extracts information from audio signal is actually at the basis of the venerable technique of the Vocoder, proposed at Bell Laboratories by Homer Dudley in the late 1920s as a device for representing the vocal signal (by means of an "encoder") and then resynthesizing it (through a "decoder" component) [22]. The main issue with a filter bank technique is that it does not respect the phase, which is crucial in signal reconstruction. Hence the digital algorithm of the Phase Vocoder that, as the name indicates, instead takes into account phase, and can be seen as a filter bank interpretation of Fourier Transform [23]. While FFT is the standard spectral tool in music information retrieval [6], filter bank approaches to frequency decomposition have been proposed as an alternative [24, 25]. Even if phase is required for signal reconstruction (i.e. at the audio level), it can be discarded in case of spectral analysis targeted at musical information extraction (i.e. sound object level). In relation to this aspect, it can be observed that the typical output in terms of musical information relies on standard music notation (the so-called common practice notation, CPN). Thus, as already discussed, if FFT data are gathered, a large data reduction has to be performed in order to output CPN, and various techniques have been proposed accordingly [6]. A

<sup>3</sup><http://earbirding.com/blog/archives/1229>

cartoonified approach to the problem can be pursued by reversing the perspective, that is, by starting from output data, i.e. pitches. While MIDI protocol encodes pitches in  $2^7$  values (0 – 127), typical musical data in CPN are satisfyingly represented by means of a piano keyboard (88 pitches). In both cases, the amount of data is fairly lower than typical FFT frequency resolutions. Following a cartoonified approach, in the SonaGraph architecture (Figure 2) the analysis step is thus performed through a bank of 88 constant-Q bandpass filters, tuned logarithmically in relation to piano keys (Log filter bank). The rationale for such a “musical” choice is to achieve a good compromise between the pitch resolution and the size of the bank, while covering more than 4 KHz (precisely, 27.5 – 4186 Hz). The filter bank has an overall tuneable resonance factor  $Q$  (therefore a variable band depending on the central frequency), on the model of the narrow/wide band distinction of the analog Sonagraph (but more general). Each filter is connected to an amplitude envelope follower (Amp follower), as in Dudley’s Vocoder encoder. The output signal resulting from amplitude following is slightly integrated (Smoother: in fact, a sort of low pass filtering) to eliminate too rapid variations. While this operation permanently compromises phase information, it provides a clearer information on amplitude variation at the sound object level (lower time resolution). Each filtered signal, once smoothed, is then converted from linear amplitude to decibel (Converter) and sampled at a regular rate (Sampler). The sampling rate ( $sr$ ) determines the spectral time resolution and can be adjusted (as in the variable speed of the metal drum in the Sonagraph). Appropriate values for sample rate depend on the analysis’ goals and on the spectral variation of the signal that is being considered: empirically, values between 10 and 50 Hz (in this last case, already at audio rate) provide a good compromise between the sound object level and audio accuracy. For each sample, the Sampler module returns a vector of 88 values estimated in dB (hence on, bin), which is stored as a column in a 2D matrix. In the latter, intuitively, rows represent the time domain, each containing the values of the sampled signal for a single filter at rate  $sr$ . The two-dimensional signal thus obtained can be analysed in real-time (Analyzer) or stored (Archivist) and imported later. With respect to the audio signal (and the FFT one), Analyzer takes into consideration an extremely reduced data amount, making the implementation of operations on the single bin –and more generally on subsets of the matrix– very simple and computationally inexpensive, thus fitting real-time operations. For example, one can easily investigate maxima and minima in spectral regions by selecting certain frequencies or obtain information about spectral peaks, e.g. ranking the first  $n$  frequencies in relation to amplitude or with respect to a threshold (i.e. frequencies with amplitude higher than  $t$ ).

The SonaGraph cartoonified analysis framework is represented in Figure 2, region A, while region B depicts some added functionalities. The frequency resolution is indeed calibrated on the 12-TET system. The latter is intended as a standard reference grid providing a uniform perceptual pitch sampling as codified by Western practice. But

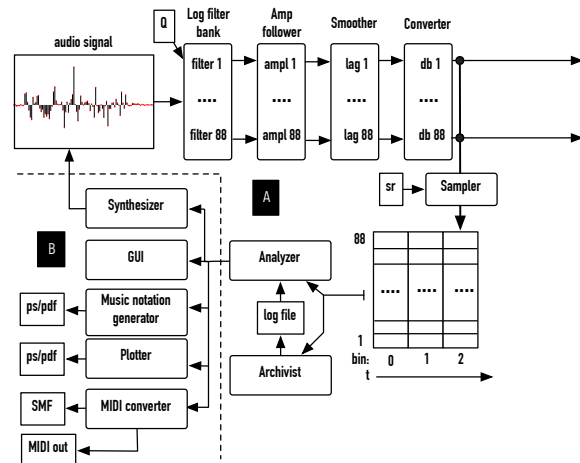


Figure 2. SonaGraph architecture.

the filter bank can be easily modified, e.g. it can be tuned by providing an array of frequencies rather than a uniform half-tone step (see [26] for a discussion on the relations between spectrum and tuning) or it can be expanded/reduced in size. As an example, to reach a quarter-tone resolution, the bank can be doubled in size in order to double resolution, or, in a non real-time fashion, two filtering processes can be run with different tunings, then gathered data can be properly assembled.

## 5. APPLICATION DESCRIPTION

The SonaGraph framework has been implemented as a set of classes for the SuperCollider audio and music programming environment [27, 28]. The latter provides sound processing capabilities via its audio server component, but also GUI programmable elements, MIDI support and a high level OOP language to manipulate spectral data, thus seamlessly bridging audio and sound object level. Moreover, other functionalities are available via internal access to the Unix terminal. All the functionalities represented in Figure 2 are encapsulated in the classes described by Figure 3: boxes represent classes while operations are indicated by labels written in plain text. Underlined text indicates the three main operation metacategories: analysis, integration/communication and graphic export.

SonaGraph is the main class, providing general audio management (including resynthesis), sampling, analysis, archiving. In relation to Figure 2 it implements section A plus Synthesizer and MIDI converter from section B. It also provides a common unified method interface for the other classes (integration, in relation to Figure 3). The filter bank is implemented via second order band pass filters, with a default (but modifiable) very high  $Q$  ( $= 1000$ ), that has proven effective. Other audio functionalities include a fundamental frequency detection analysis (implementing the Tartini algorithm [29]), synced to the same sampling rate of the Sampler, and a bank of sine oscillators and a virtual piano instrument, both to resynthesize spectral data as a control step (Synthesizer in Figure 2). SonaGraph also includes MIDI support, both in terms of real-time MIDI



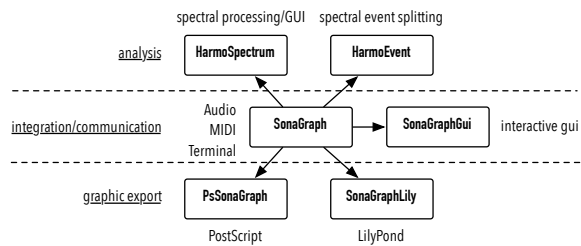


Figure 3. SonaGraph classes.

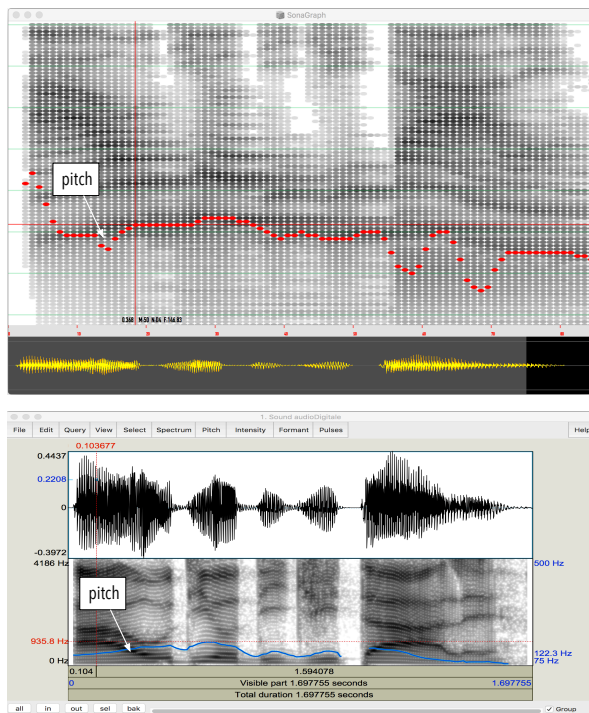


Figure 4. SonaGraph interactive GUI (top) and Praat (bottom).

communication (e.g. to MIDI synthesizers, as those supported by DAWs) and Standard MIDI File (SMF) creation. The SMF export creates voices by selecting rows in the sonogram. In order to simplify the MIDI file structure, a voice grouping algorithm is applied, so that consecutive pitches having an amplitude greater than a selected threshold are grouped together in a single note.

The SonaGraphGui class supports an interactive GUI for inspection and playback of the analyzed sound. Its aim is to help the exploration of gathered data both visually and aurally. Mainly inspired by Praat GUI<sup>4</sup>, it provides a scalable window showing the sonogram (top) and the waveform (bottom), and including also a visualization for the estimated fundamental pitch. Figure 4 shows the SonaGraph GUI (top) and the Praat GUI (bottom) for the same signal (a voice sample: formants are apparent) for sake of comparison. In the SonaGraph GUI, mouse-pointing in the window results in a vertical red line indicating the selected bin and a horizontal one showing the frequency.

<sup>4</sup> <http://www.praat.org/>

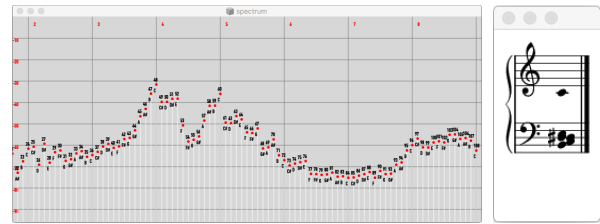


Figure 5. HarmoSpectrum GUI.

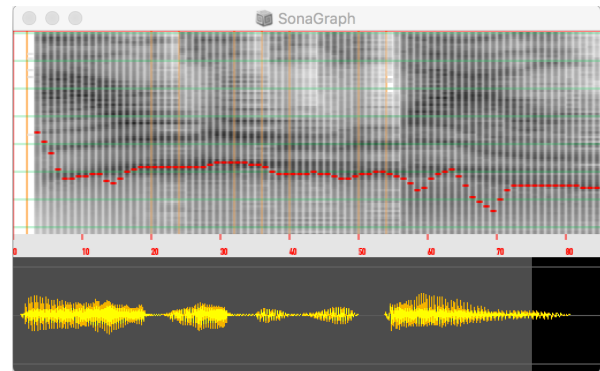


Figure 6. HarmoEvent onset data in the GUI.

On the left of the vertical line, time is indicated (in Figure 4: 0.368), on the right MIDI pitch (50), note (D4) and frequency (146.83) are shown for the selected point. Space bar allows to start/stop source playback from the selected bin. While clicking, a synthesized short piano note is generated to provide a reference for the pointed pitch. A threshold for amplitudes can be set, so that values under the threshold are not shown. In the SonaGraph GUI, pitches in red indicate the fundamental pitch estimation (like the blue line in the Praat GUI, in both cases a label has been added to the GUIs in Figure 4 to help the reader). Note that formants are highly visible in a speech sound (and help comparison between the GUIs), but they do not coincide with the fundamental pitch.

The HarmoSpectrum class features all the methods for spectrum processing, including interactive GUI and music notation generation. Conceptually, it operates on a single bin while operations on multiple bins (i.e. selecting a bin range representing more time samples) are handled by SonaGraph by averaging amplitudes and passing the averaged single bin to HarmoSpectrum. Data are available to the SuperCollider language for further manipulation and can be explored visually through an interactive dedicated GUI (Figure 5, left). The GUI shows the spectral envelope for the selected, averaged spectrum (bins 40 to 50 from Figure 4, left): each component is labelled with MIDI note (top) and symbolic name (bottom), while octaves are indicated by vertical lines. By clicking on the window, a piano note is played back for the selected pitch as a reference. As it can be seen from spectral peaks in Figure 5, left, the most prominent pitches in MIDI notation are 47, 48, 50, 51, 52, 60 (49 is just under the selected threshold). Given an amplitude threshold, they can be viewed directly in musical





Figure 7. Generated notation imported into GUI.

notation by invoking a dedicated method that exploits the LilyPond environment for notation description and typesetting [30]. The method writes LilyPond code on a temporary file, renders it by calling the LilyPond executable via Unix terminal, and loads the rendered image into a window (Figure 5, right). By clicking on the window, the chord is played back using a synthesized piano. The LilyPond code can be written on a user-specified file so to remain accessible for further usage.

HarmoEvent performs some basic operations on sonogram evolution over time in order to recognize discontinuities. As the only information is the spectral one, it detects an “event” by comparing the difference between averaged amplitudes of adjacent bins. Two parameters are available: a threshold value for minimum amplitude difference, and minimum distance (in bins) between events. While very crude, this operation implements a typical approach to on-set detection by spectral-based novelty function (or spectral flux, [6]). Once detected, events can be automatically visualized in the GUI (see vertical orange lines in Figure 6, that also demonstrates different displaying threshold and ratio with respect to Figure 4). A segmentation procedure is available that split the signal between adjacent onsets, thus obtaining event sub-signals: it applies a minimum envelope to avoid clicks, and exports the resulting event signals to audio files.

As already discussed, CPN visualization and export are crucial in bridging spectral content to music application. Following the model discussed for HarmoSpectrum, SonaGraphLily manages the mapping from sonographic data (rather than spectral snapshots) to music notation by generating LilyPond code. It creates LilyPond text source files, renders them as graphic files and –if needed– loads them into GUI for real-time playback. Automatic generation of notation is a complex topic, and various heuristic approaches have been proposed [31, 32]. In our case, the threshold setting for filtering out lower amplitudes is

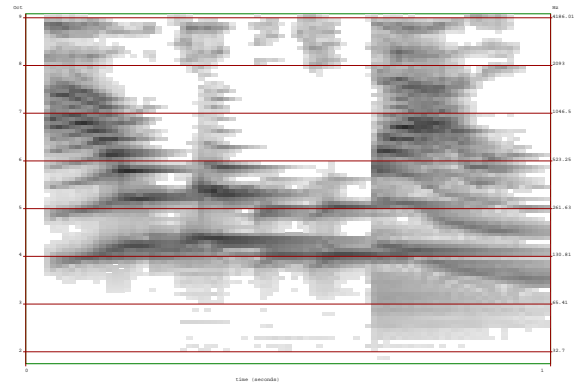


Figure 8. A PostScript rendering.

crucial in order to avoid cluttered notation. Two solutions are provided by the SonaGraphLily class. In the first case, voices are created from sonogram pitch rows and displayed on separate staves. While this visualization is useful from an analytical point of view, the number of voices may escalate quickly (with default values, up to 88), thus becoming visually unmanageable. A second solution groups notes according to a standard piano notation, with much more (but potentially too much) compact results. For sake of comparison, Figure 7 shows for the same audio sample (again, the one from Figure 4) the two notations as loaded in the GUI. As in SMF export, a voice grouping algorithm is applied (not applicable in Figure 7). The voice-based notation window (Figure 7, top) has been cut to 7 voices (of 16) for sake of readability. The transcription algorithm transparently maps the sonogram’s time resolution by assigning a semiquaver duration value to each bin. Tempo is thus calculated by taking into account the sample rate (here,  $12\text{ Hz} = 12/4 \times 60 = 180\text{ bpm}$ ), while meter is set to 4/4. In Figure 7, the rendered notation files are loaded into a GUI, that allows for playback using both synthesized piano (for note data) and the original audio sample (as a comparison), providing also a crossfade slider for variable mixing (spec/snd, spectrum vs sound).

Finally, the PsSonaGraph class is dedicated to graphical export of the sonogram, using the standard (but customizable) grey scale for plotting amplitudes of frequencies over time. It creates a PostScript file [33] from analysis data with adjustable graphic parameters, and converts it into PDF format via terminal utilities. Figure 8 shows a PostScript output from the sonogram in Figure 4, by setting a higher amplitude threshold. It includes reference octave and frequency annotations at each side of the red lines.

## 6. A SHORT ANALYTICAL COMPARISON

Although cartoonified, to an informal perceptual appreciation the frequency resolution of SonaGraph provides useful and adequate clues on the spectral information of the sound taken into account, even if the model per se is undoubtedly oriented specifically towards harmonic information. In order to further verify the results, in this section two automatic transcriptions from spectral data over time

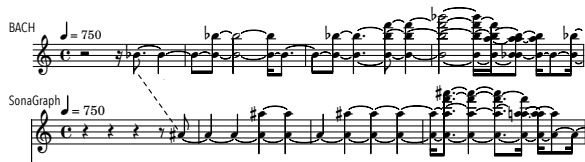


Figure 9. Music notation transcription for “trumpet”.

into musical notation are compared. In particular, the comparison has been performed between SonaGraph and Audiosculpt<sup>5</sup> /BACH [12]. In Audiosculpt, FFT data are processed via partial tracking, in order to filter out amplitudes lower than a selected threshold. The resulting SDIF file is imported into the BACH library for Max/MSP that allows for pitch interpretation of spectral data. As BACH is a state-of-the-art tool in assisted composition, it shares with SonaGraph the same purpose and intended users. Four mono files have been taken into account. With the aim of comparing the two systems, samples have been chosen to represent spectral configurations with different features in relation to different acoustic situations (harmonic/inharmonic, monophonic/polyphonic, music/environment). All sound files have been normalized previously to the analysis processes. The analysis by SonaGraph has been performed according to the previous discussion, with a sample rate = 50 Hz and by selecting all amplitudes > -30 dB. Such a sample rate results in a very high tempo = 750 bpm, as the rationale in this case (rather than in Figure 13) is not to provide a performance indication but to capture spectral transformation. In BACH, a time quantization is introduced, so that it matches the SonaGraph’s one for sake of comparison, both in terms of tempo, note value quantization (16th) and meter (4/4). BACH’s pitch resolution has also been constrained to half-tones (standard MIDI notes). As it is possible to export from BACH music notation as LilyPond code, results can be easily compared (even if BACH export uses a single treble clef, while SonaGraph a piano staff). Figures 9-12 show the transcriptions for four audio samples. Amplitude threshold for SonaGraph example is kept at -30 dB, while Audiosculpt/BACH threshold is adjusted so to provide comparable examples. As apparent from the examples, BACH and SonaGraph makes use of two opposite enharmonic transcription strategies, respectively assigning  $\flat$  and  $\sharp$  alterations. Temporal misalignment in terms of notation results from implementation details, depending on a fixed offset at initialization, and, if relevant, is indicated by a dashed line.

Results are substantially the same if a harmonic spectrum is taken into account, as in the case of a melodic trumpet phrase (Figure 9). Figure 10 shows the transcription of a sample from a wind turbine, presenting some harmonic components over a very noisy background. In this case, while generally coherent, transcriptions have proven to strongly depend on amplitude threshold. The SonaGraph’s one (bottom) is strongly sensitive to some higher frequency components that characterize the sound attack. They can

<sup>5</sup> <http://anasynt.h.ircam.fr/home/english/software/audiosculpt>



Figure 10. Music notation transcription for “wind turbine”.



Figure 11. Music notation transcription for “octandre”.

be revealed in the Audiosculpt analysis by lowering the threshold, but many other frequencies then become relevant for transcription. This clearly shows that the two analysis model have a different sensitivity. The same situation applies to Figure 11, a transcription from an excerpt from dense orchestral sound (a tutti in *ff* from Varèse’s *Octandre*). While the overall material is approximatively the same, sensitivity to amplitudes varies between the two analysis. A general difficult case for spectral analysis is related to noisy sounds. Figure 12 shows a transcription from a coin tossed on a hard surface, with no clear harmonic content. While both transcriptions capture a generic energy accumulation in the same higher frequency region, details in terms of pitch sensibly vary.

In conclusion, transcriptions in both environments are substantially coherent, sometimes revealing more or less clearly various perceptual details, as a result of the different sensitivities to amplitude. While the BACH system is generally more flexible, the automatic notation, relying on FFT, is generated by a hidden process, not directly accessible to the musician. On the other side, SonaGraph data, while simplified, are directly mapped into notation, and their manipulation by the user can be transparently observed into it.

## 7. MUSICAL EXAMPLES

A first straightforward musical application of the SonaGraph framework has occurred in the piece/installation *Orologio da rote*<sup>6</sup>. The piece is a reflection on signals broadcast in the Italian mediascape and includes a set of music quotations from historical jingles from RAI, the Italian national public broadcast service. It is scored for 3 modified radios and an automated piano, in particular a Yamaha Disklavier that can be driven by MIDI messages. Once collected from various sources, both acoustic signals and music jingles have been analyzed via SonaGraph and the resulting spectral data stored. During the performance, data are converted into MIDI, and MIDI messages sent in real-time to

<sup>6</sup> <https://soundcloud.com/vanderaalle/orologio-da-rote>

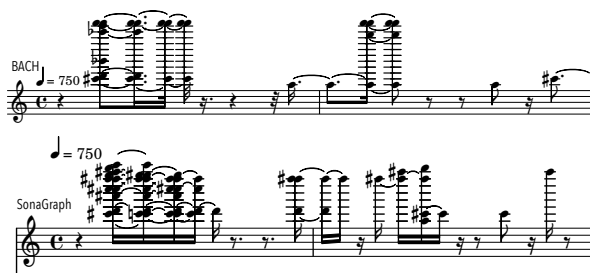


Figure 12. Music notation transcription for “coin”.

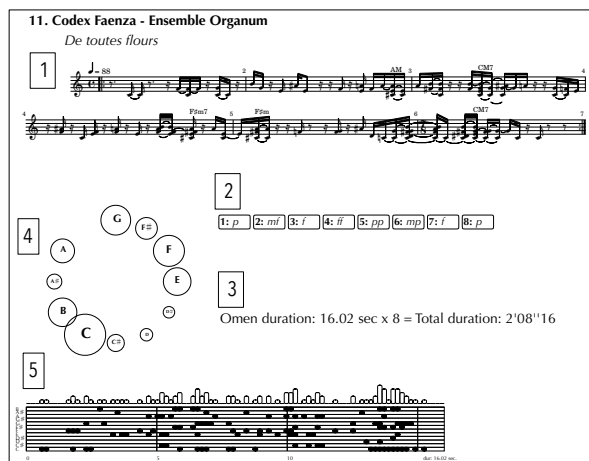


Figure 13. Omen notation for *Haruspex cledonomans*, 11.

the Disklavier (with amplitudes mapped onto velocities). Most of the piano material is thus a sort of spectral (and ghost-like) reconstruction of existing sounds, not dissimilarly from Peter Ablinger’s *Speaking piano*<sup>7</sup>.

A second project has involved SonaGraph in composition and heavily relies on music information retrieved from gathered data. The work *Haruspex cledonomans* is a collection of “ominous formulas” for improvisation written for *ad libitum* improvisers<sup>8</sup>. Each piece of the collection is a formula that describes information for two logical music layers, the “omen”, written in standard music notation, and the “prophecy”, to be constructed/improvised while the omen is playing. Omens originate from 43 short audio phrases extracted from recordings of various composers/musicians, including classic, jazz, rock and ethnic sources. All the fragments have been analyzed via SonaGraph. In each analysis, the sample rate has been matched by ear to be approximatively synchronized to music pulse. Then, the spectral data have been automatically transcribed into musical notation, disregarding octaves so that only pitch classes are present (i.e. chroma). Omens are intended as the background layer for improvisation, and in each piece information extracted from the sonogram is provided to the musician(s) as a guide for the “prophecy” improvisation. As a consequence, some basic MIR techniques have been applied to SonaGraph analysis data. Figure 13 shows one of

the omens. The whole notation is generated automatically from processed spectral data by means of LilyPond (as described before) and the Python-based Nodebox graphic environment<sup>9</sup>, with Python code scripted from SuperCollider. In Figure 13, the five blocks provide various information for the improviser. While blocks 2 and 3 depend on different compositional parameters (here not relevant), blocks 1, 4 and 5 are directly generated from SonaGraph data. Block 1 provides the omen notation in CPN (as discussed before). While the omen has to be played by the background musician, notation also includes a possible chord interpretation of note clusters (i.e. simultaneous collections of more than 2 notes), intended as a guide for the improviser. An analysis step is performed by a specialized class, not yet included in the framework core, ChordAnalyzer. Chords are specified in a template list collecting abstract chord structures (e.g. a major chord is indicated as {0,4,7}). Templates have been compiled from various music theory sources. If a cluster is found and if it matches a certain chord structure, then the relevant chord indication is written on top of the staff. Chord analysis works enharmonically and disregards chord position. Block 4 (“Pitch class relevance”) displays the chroma set with symbolic names, varying the font and the circle sizes proportionally to the amount of occurrences of each pitch class in the sonogram. It thus indicates to the improviser possible pivot notes to be taken into account. Finally, block 5 is a visualization of the omen as a piano roll. It allows to quickly understand chroma distribution over time. Time is proportional to duration, so that the piano rolls of the various ominous formulas are scaled proportionally to their absolute durations. On top, a histogram provides an overall indication of the number of pitches for that time unit (“amount”), as a general density information.

## 8. CONCLUSIONS

The SonaGraph framework proposes a cartoonified (i.e. simplified but effective) model for spectral analysis oriented toward computer-assisted and algorithmic composition. Geared toward symbolic applications, it extracts spectral information that, while strongly reduced, is still adequate perceptually. Such a reduction makes the model efficient in terms of data storage and manipulation, and suitable for real-time usage. Thus, it can be easily integrated into a pipeline connecting sound to music application, in terms of symbolic representation (music notation and MIDI). In short, while sketchy, the gathered data are transparent to music manipulation (sound object level) rather than to sound (audio level). The SuperCollider code of the actual implementation (still in progress) is available on GitHub<sup>10</sup> and includes help files with examples.

## Acknowledgments

The author is grateful to Luca Morino for providing examples in BACH in relation to section 6.

<sup>7</sup> [https://ablinger.mur.at/speaking\\_piano.html](https://ablinger.mur.at/speaking_piano.html)

<sup>8</sup> <https://soundcloud.com/vanderaalle/sets/haruspex-cledonomans>

<sup>9</sup> <https://www.nodebox.net/code/index.php/Home>

<sup>10</sup> <https://github.com/vanderaalle/vanderaalleSC/tree/master/sonaGraph>

## 9. REFERENCES

- [1] C. Roads, *Composing electronic music. A new Aesthetic*. Oxford: Oxford University Press, 2015.
- [2] E. R. Miranda, *Computer Sound Design*. Oxford: Focal Press, 2001.
- [3] S. Canazza, G. De Poli, G. Antonio Mian, and A. Scarpa, "Real time comparison of audio restoration methods based on short time spectral attenuation," in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-01)*, Limerick, 01 2001, pp. 1–4.
- [4] J. Bresson, "Sound Processing in OpenMusic," in *Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06)*, Montreal, 2006, pp. 325–330.
- [5] A. Agostini, É. Daubresse, and D. Ghisi, "Cage: a High-level Library for Real-time Computer-aided Composition," in *Proceedings ICMC—SMC—2014*, A. Georgaki and G. Kouroupetroglou, Eds., Athens, 2014, pp. 308–313.
- [6] M. Müller, *Fundamentals of Music Processing. Audio, Analysis, Algorithms, Applications*. Cham: Springer, 2015.
- [7] P. Schaeffer, *Traité des objets musicaux*. Paris: Seuil, 1966.
- [8] C. Roads, *The Computer Music Tutorial*. Cambridge, MA, USA: MIT Press, 1996.
- [9] C. Cannam, C. Landone, M. Sandler, and J. Bello, "The sonic visualiser: A visualisation platform for semantic descriptors from musical signals," in *ISMIR 2006 - 7th International Conference on Music Information Retrieval*, 2006, pp. 324–327.
- [10] D. Rocchesso and F. Fontana, Eds., *The Sounding Object*. Firenze: Edizioni di Mondo Estremo, 2003.
- [11] A. Klapuri and M. Davy, Eds., *Signal Processing Methods for Music Transcription*. New York: Springer, 2006.
- [12] A. Agostini and D. Ghisi, "A Max Library for Musical Notation and Computer-Aided Composition," *Computer Music Journal*, vol. 39, no. 2, pp. 11–27, 2015.
- [13] A. Schneider, *Sound-Perception-Performance*, ser. Current Research in Systematic Musicology. Cham: Springer, 2013, ch. Change and Continuity in Sound Analysis: A Review of Concepts in Regard to Musical Acoustics, Music Perception, and Transcription, pp. 71–111.
- [14] P. N. Lehner, *Handbook of Ethological Methods*. Cambridge: Cambridge University Press, 1996.
- [15] R. Jakobson, C. M. Fant, and M. Halle, *Preliminaries to Speech Analysis. The Distinctive Features and their Correlates*. Cambridge, Mass.: The MIT Press, 1952.
- [16] C. M. Fant, "Historical Notes," *TMH-QPSR*, vol. 47, pp. 9–19, 2005.
- [17] É. Leipp, *Musique et acoustique*. Paris: Masson, 1971.
- [18] A. Orcalli, *Fenomenologia della musica sperimentale*. Potenza: Sonus, 1993.
- [19] C. Regnault, "From quantitative to qualitative. the pertinence of sonographic representation for soundscape analysis," in *Proceedings of inter.noise 2000*, 2000, pp. 1–5.
- [20] F.-B. Mâche, *Musique, mythe, nature ou Les dauphins d'Arion*. Paris: Klincksieck, 1983.
- [21] J. T. Marshall, "Voice in communication and relationship among brown towhees," *The Condor*, vol. 66, no. 5, pp. 345–356, 1964.
- [22] H. Dudley, "The Carrier Nature of Speech," *The Bell System Technical Journal*, vol. XIX, no. 4, pp. 495–515, 1940.
- [23] M. Dolson, "The Phase Vocoder: A Tutorial," *Computer Music Journal*, vol. 10, no. 4, pp. 14–27, 1986.
- [24] M. Müller, *Information Retrieval for Music and Motion*. Heidelberg: Springer, 2007.
- [25] C. Schörkhuber and A. Klapuri, "Constant-Q Transform Toolbox for Music Processing," in *Proceedings of 7th Sound and Music Computing Conference*, X. Serra, Ed. Barcelona: SMC, 2010.
- [26] W. Sethares, *Tuning, Timbre, Spectrum, Scale*. London: Springer, 2005.
- [27] S. Wilson, D. Cottle, and N. Collins, Eds., *The Super-Collider Book*. Cambridge, Mass.: The MIT Press, 2011.
- [28] A. Valle, *Introduction to SuperCollider*. Berlin: Logos, 2016.
- [29] P. McLeod and G. Wyvill, "A smarter way to find pitch," in *Proceeding of the 2005 International Computer Music Conference*, X. Serra, Ed., Barcelona, 2005, pp. 138–141.
- [30] H.-W. Nienhuys and J. Nieuwenhuizen, "LilyPond, a system for music engraving," in *Proceeding of the XIV CIM 2003*, Firenze, 2003, pp. 167–172.
- [31] D. Byrd, "Music notation software and intelligence," *Computer Music Journal*, vol. 18, no. 1, pp. 17–20, 1994.
- [32] A. Valle, "Integrated Algorithmic Composition. Fluid Systems for including notation in music composition cycle," in *NIME 2008: Proceedings*, 2008, pp. 253–256.
- [33] Adobe, *PostScript Language Reference*, 3rd ed. Reading, Mass.: Addison-Wesley, 1999.



# Sound in Multiples: Synchrony and Interaction Design using Coupled-Oscillator Networks

Nolan Lem

Center for Computer Research in Music and Acoustics (CCRMA) Stanford University  
nlem@ccrma.stanford.edu

## ABSTRACT

Systems of coupled-oscillators can be employed in a variety of algorithmic settings to explore the self-organizing dynamics of synchronization. In the realm of audio-visual generation, coupled oscillator networks can be usefully applied to musical content related to sound synthesis, rhythmic generation, and compositional design. By formulating different models of these generative dynamical systems, I outline different methodologies from which to generate sound from collections of interacting oscillators and discuss how their rich, non-linear dynamics can be exploited in the context of sound-based art. A summary of these mathematical models are discussed and a range of applications are proposed in which they may be useful in producing and analyzing sound. I discuss these models in relationship to one of my own kinetic sound sculptures to analyze to what extent they can be used to characterize synchrony as an analytical tool.

## 1. INTRODUCTION

Coupled Oscillators networks are dynamical systems that describe how ensembles of interacting elements are able to self-organize and synchronize over time. In terms of sensory perception, they have been examined in a wide range of fields including those related to rhythmic entrainment, biomusicology, psychoacoustics, signal processing, and generative music [1, 2]. In the field of computer music, there have been a plethora of synthesis techniques that attempt to generate interactive and collective phenomena. These include techniques related to additive and granular synthesis, microsound, swarm models, texture synthesis, physical modeling synthesis, and statistical signal processing [3–5]. Previous work in coupled oscillators as a generative musical devices has been explored by Lambert where he looks at coupled oscillators as a "stigmergic" model, producing complex output through an audience's interaction with a system of coupled Van der Poll oscillators [6]. Operating within a similar territory, this paper proposes several generative paradigms to create sound in different

synthesis and rhythmic schemes. Lastly, I describe one of my own kinetic sound sculptures that was inspired from the system dynamics of a specific coupled oscillator model.

## 2. MATHEMATICAL DESCRIPTION

Coupled oscillators are a broad category of interacting dynamical systems that describe a wide range of natural phenomena such as firefly synchronization, pace maker cells, neural networks, and cricket chirping models [7, 8]. One of the most basic coupled oscillator models is known as the Kuramoto model [9]. In this formulation, the governing equation for each oscillator's phase is shown for the ensemble in Equation (1)

$$\dot{\phi}_i = \omega_i + \frac{K}{N} \sum_{j=1}^N \sin(\phi_j - \phi_i - \alpha_o) \quad (1)$$

where  $\phi_i$  is the phase of the  $i_{th}$  oscillator and  $\dot{\phi}_i$  is the derivative of phase with respect to time.  $\omega_i$  is the intrinsic frequency of the oscillator,  $i$ , in a population of  $N$  oscillators.  $K$  is the coupling factor and the  $\sin(\phi_j - \phi_i)$  term is the phase response function that determines the interaction between each oscillator and the group. We can add a phase offset or "frustration" parameter  $\alpha_o$  in the phase response function to force oscillators into different phase orientations or to account for a time delay in the model.

As a visual description, it's useful to describe the system by the movement of a "swarm of points" moving about a circle, each point representing one oscillator with its own intrinsic frequency drawn from a probability distribution,  $g(\omega)$  (which is generally taken to be a unimodal gaussian distribution).

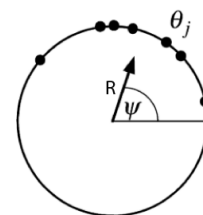


Figure 1. "Ensemble of coupled oscillators represented in a circle map as a "swarm of points" moving about a circle [7].

Copyright: © 2019 Nolan Lem et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Depending on the  $g(\omega)$  from which the oscillators are drawn, Kuramoto was able to show that in the limit as  $N$  goes to  $\infty$ , the critical coupling,  $K_c$ , will define the point at which the system will undergo a phase state transition characterized by collective synchrony. This critical coupling,  $K_c$  is shown in Equation (2)

$$K_c = \frac{2}{\pi g(0)} \quad (2)$$

where  $g(0)$  is the mean of the distribution of initialized intrinsic frequencies in the set of  $\omega_i$ . If  $K > K_c$  the oscillators' phases will begin to spontaneously align and the system state can be said to be characterized by synchrony.

We can extract the complex order parameters,  $R$  (phase coherence) and  $\psi$  (average phase) to solve for the system in the limit as  $N$  goes to  $\infty$ . This modifies the governing equation to be in terms of a mean-field approximation of the oscillators' phases: each oscillator is no longer beholden to the phase of every other oscillator but is coupled to the ensemble's summed, average phase. This is shown in Equation (3).

$$Re^{j\psi} = \frac{1}{N} \sum_{i=1}^N e^{j\psi_i} \quad (3)$$

The phase coherence  $R$  is a good indication of the synchrony of the system at large: when  $R = 1$  the system exhibits complete synchrony (all phases are aligned) and when  $R = 0$ , the oscillators are desynchronized (points are simply running around the circle at their own intrinsic frequency,  $\omega_i$ ). Applying these complex order parameters to Equation (1), we form Equation (4).

$$\dot{\phi}_i = \omega_i + KR \sum_{j=1}^N \sin(\psi - \phi_i) \quad (4)$$

We can add an external forcing term by adding another term with a different phase response function,  $\Lambda_e(\phi_i)$  as shown in Equation (5).

$$\dot{\phi}_i = \omega_i + \Lambda_e(\phi_i) + KR \sin(\psi - \phi_i) \quad (5)$$

Now the system equations demonstrate a trade-off between frequency alignment by external forcing and phase alignment by the attractive coupling as a function of their phase response curves. We can choose  $\Lambda(\phi)$  to be from any distribution but certain functions are associated with different system behavior. For example, if we let  $\Lambda$  be a "sawtooth interaction function" [10], we can force the oscillators into a "incoherent state" where all oscillators will settle on the same frequency but with a constant phase offset,  $\alpha_o$ , as seen in Equation (1). Depending on  $N$ , this will space out the oscillators to have a constant phase offset,  $0 < \phi_c < 2\pi$ .

More complex behavior can emerge when we let  $\omega_i$ ,  $K$ ,  $N$  and  $\Lambda(\phi)$  of Equation (5) become a function of time as well. Additionally, even more complex behavior arises

when we let  $K$  take on different values between different micro-ensembles of coupled oscillators.

The complex order parameters are simply one way to evaluate the group synchrony of the system. Frank and Richardson's "cluster phase method" uses the complex order parameters to derive another degree of synchronization in multi-variate time series [11]. These have implications in different sonification, synthesis and rhythmic schemes that result from the aforementioned generative model.

### 3. COUPLED OSCILLATORS AS GENERATIVE SONIC DEVICES

Using this coupled oscillator model, we can extend these different parameters and states to synthesize sound on a continuum of collective rhythms both at the beat and sample level. As a general paradigm, rather than solving these  $N^{th}$  order equations analytically (which computationally can become intractable rather quickly), we can generate the system output using numerical analysis and employ it to generate sound in several different ways. As such, we can modify the rate at which the system is generated and map the output to sonic parameters in different perceptual time scales. This is the crucial link that maps a theoretical mathematical system to the sensory phenomena of auditory processing. For this end, this approach can be looked at as a sonification of the data that operates on a temporal spectrum.

#### 3.1 Sound synthesis with Coupled Oscillators

##### 3.1.1 Additive Synthesis

Because these non-linear oscillators trace out sinusoidal trajectories, the most basic synthesis method would be to simply treat the system as an oscillator bank where each oscillator's instantaneous phase is a signal amplitude at an audio rate. As the system begins to self-organize and phase-align, their collective entrainment would be perceived as a collection of sine waves of different initial frequencies emerging to a single frequency over time. As the coupling coefficient is increased to reach  $K_c$ , Fig. 2 shows the power spectrum of a group of oscillators becoming entrained to the center frequency of a gaussian distribution of oscillators from 0 to 5 kHz. Here we can see how oscillators with intrinsic frequencies near the center of frequency distribution are recruited (or entrained) first whereas oscillators near the tails of the distribution take longer to synchronize to the mean frequency.

We can also replace the external forcing in Equation (5) with the frequency content of audio that could drive the individual oscillators. In this synthesis model, the oscillators could act as a  $N_{th}$  order filter bank with center frequencies determined by their assignable intrinsic frequencies. This differs from a phase vocoder model insofar as the center frequencies of the filter bank are not fixed in frequency but are coupled according to some schema and therefore allowed to deviate by some amount.

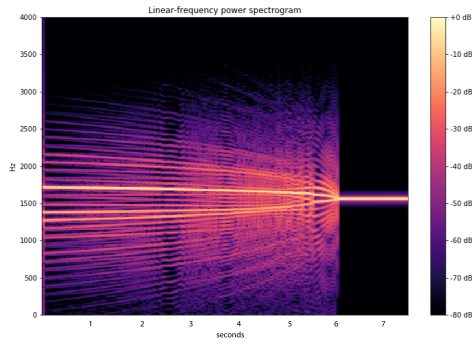


Figure 2. Ensemble of 100 sinusoidal oscillators becoming entrained to a frequency at the center of the distribution.

### 3.1.2 Spectral Processing

Other more complex synthesis techniques can be derived from extracting instantaneous phase from an input audio file using instantaneous frequency estimation techniques where the signal can be decomposed into a collection of instantaneous phases (via a Hilbert Transform and phase unwrapping). We can apply coupling between the instantaneous phases using coupled oscillator dynamics to perform transformations on the temporal or spectral information. For example using a FFT interpretation of the phase vocoder model, we can divide the time-varying signal into several spectral bands and—after unwrapping each channel’s instantaneous phase—apply band-limited coupled oscillator networks to modulate their instantaneous phase forcing them to become entrained to a center frequency within the spectral band over time. Because the critical coupling of Equation (2) is a function of the intrinsic frequency distribution set by  $g(\omega)$ , we can populate these spectral regions of the input spectrum with oscillators drawn from a gaussian distributions with  $\mu$  centered around the FFT bin center frequency. This has the effect of encouraging oscillator synchronization within the channel-dependent (band-limited, constant-Q) region. In this synthesis scheme, the external forcing function,  $\Lambda_e(\phi_i)$ , is passed the instantaneous phases extracted from each of the spectral bands by the FFT. An example is shown in Fig. 3 where a flute playing a major scale is resynthesized using the aforementioned method. This example makes use of 130 oscillators split into 10 coupled groups where coupling is increased over the duration of the sound file ultimately resulting in full synchrony per band-limited group.

Ultimately, this coupled-oscillator phase vocoder model would allow the frequency content of an input audio signal to modulate and synchronize the frequency content (or spectral entrainment) of the source sound. Sounds that are characterized by spectra that conforms to certain harmonic relationships could force the coupled oscillators into different periodic or synchronous states. Clearly, because this method utilizes phase vocoding analysis, it would work

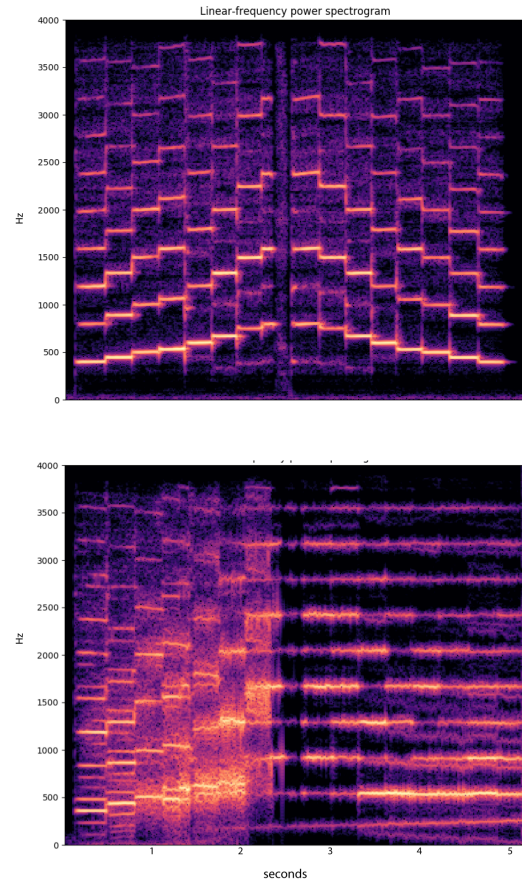


Figure 3. Spectrum of flute playing a major scale (top). Phase vocoder coupled oscillator resynthesis using ten coupled oscillator networks (bottom).

best with analysis techniques that prioritize horizontal phase coherence over vertical phase coherence.

### 3.2 Rhythmic Generation: Coupled Oscillators as Control Signals

We can use the dynamics of the coupled oscillator system to control rhythmic generation or musical parameters. The idea of synchronization lends itself well to many aesthetic ideas of minimalist and procedural music where musical parameters are slowly modulated over time.

If we set the coupled oscillator ensembles to be iterated at a rate that is well below a sampling rate suitable to audio synthesis, we can use Equation (5) to trigger audio events when the instantaneous phase of each oscillator  $\phi_i$  encounters a zero-crossing. To accomplish this, we can trigger an "audio event" using the basic sonification scheme detailed in Equation (6) below.

$$\text{audio event}(\phi_i) = \begin{cases} 1, & \text{if } \phi_{i-1} < \phi_i \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Therefore, as each oscillator completes one cycle (crosses the zero-point of the circle), they trigger a sound such as the playback of a sample. The system generates complex rhythmic behavior when different groups of oscillators take on different coupling coefficients to form microensembles that are locally coupled. When the system parameters are modulated over time, the system can be forced into different polyrhythmic relationships that converge and devolve over time.

This could also be meaningfully applied to musical forms by allowing the instantaneous phase of each oscillator to control the position of a virtual "playback" head of a rhythmic figure to create complex temporal canons that can be brought together in temporal unison by adjusting the coupling coefficients over time. Similarly in the realm of synthesis, we could allow the instantaneous phase to control the playback (or the index of a buffer) of a sampled sound file in a buffer. In this paradigm, the system produces control signals to modulate parameters of a piece of music.

#### 4. SONIC PHENOMENA AND COUPLED OSCILLATORS

If these mathematical models are sufficiently generalizable and applicable to musical analysis, they can describe and generate a plethora of meaningful musical techniques with examples taken from contemporary music composition and sound art. Perhaps the most axiomatic example demonstrating collective perceptual entrainment is Györgi Ligeti's *Poème Symphonique* (1962) for 100 metronomes [12]. In this piece one-hundred metronomes are pre-wound, set to different tempos, and then triggered en masse. As each metronome comes to rest at different times, the dynamics of the ensemble at large are well modeled using a uncoupled oscillator model where each metronome is set to a different  $\omega_i$ . As different auditory streams of periodic rhythm are presented concurrently, the listener latches onto different frames of temporal reference where a sense of beat (induction) emerges from their competing periodic stimuli. Modifying Ligeti's original piece by coupling the metronomes by placing them onto a low-friction surface such as a table with wheels, the mechanical movements of the pendulums will begin to couple their swinging motion to one another. If coupling is sufficiently strong, the metronomes will become phase-aligned to tick at a mean frequency [13].

The simultaneity of periodic rhythms characterized by *Poème Symphonique* can also be well applied to the analysis of acoustic crowd dynamics where researchers have used coupled oscillator models with spatial mean-field coupling to account the physics of crowd applause [14]. This acoustic phenomena bears resemblance to many stochastic generative methods that are capable of modeling the sound of natural phenomena (e.g. rain, hail, wind, etc.). However, the potential of the system to be controlled to self-organize over time might allow for interesting forms of collective synchrony that emerge amidst the dense acous-

tic textures characterized by nature. In terms of acoustic signalling in animal populations, coupled oscillator systems have been used to describe many different forms of biomusicological phenomena particularly those related to chorusing and stridulation [1, 15, 16]. Using research from these biomusicological models, generative chorusing synthesis that incorporate coupled oscillator synchronization methods could be an interesting avenue of exploration in sound generation and user interface design.

##### 4.0.1 Compositional Techniques

In terms of music analysis and composition, coupled oscillator dynamics of synchrony can be thought of as a temporal canon in which different fugal patterns are stretched and compressed over time to conform to a governing temporal duration. In the minimalist genre, the rhythmic "phasing" effect in Steve Reich's music (e.g. "Clapping Music", "Come Out", "Piano Phase") could be approximated by a coupled oscillator model that converges in and out of synchrony. "Phasing" could be accomplished by setting an ensemble of oscillators with different initial phases but the same intrinsic frequencies and phase-aligning them over time. Reich himself has intuited that in this compositional technique, "[t]he listener becomes aware of one pattern in the music which may open his ear to another, and another, all sounding simultaneously and in the ongoing overall texture of sounds." [17]. His formulation of pattern as rhythm reinforces similar perceptual notions of Ligeti's *Poème Symphonique* insofar as that the listener has access to simultaneous layers of competing perceptual information and that auditory feedback allows certain phenomena to take precedence over others.

Lastly in the field of sound-based art, several contemporary artists have experimented with auditory phenomena that is well-modeled by coupled oscillator systems. These include works by Zimoun, Pei Lang, and Céleste Boursier-Mougenot [18]. These artists are known for their use of multiples of sound objects set in repetitive motion to create large masses of sound from simple additive means. For instance, Zimoun's installation-based work employs hundreds of kinetic objects to construct complex sound masses in physical environments. Taken to the extreme, these sound sculptures make use of a material-oriented additive synthesis that could be approximated by dynamical system models.

#### 5. MODELLING SYNCHRONY THROUGH SCULPTURAL FORM: HIVEMIND

The rich musical dynamics inherent in coupled oscillator networks have inspired my own sonic investigations in an attempt to experiment with how to exploit these systems in physical, sculptural form. Much of my understandings of coupled oscillator dynamics in sound have been through the development of computational models that have allowed me to interact with this dynamical system through mathe-



474 mathematical analysis and numerical analysis<sup>1</sup>. For these pieces, I've written programs to explore numerical analysis (python), user interaction in real-time (SuperCollider), and performance based programs (CHuCK) to allow me experiment with the system behavior under different parameterizations. This repository also hosts several synthesis implementations mentioned in section 3.1.

### 5.1 Audio-visual Resonance: "Hivemind"



Figure 4. "HiveMind" at Pioneer Works Brooklyn, NY. video: <https://vimeo.com/127874298>

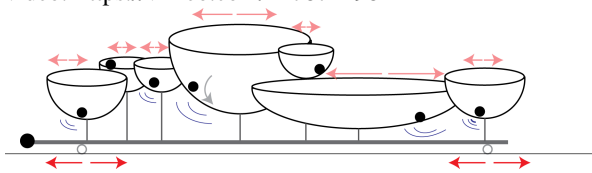


Figure 5. Kinetics of audio-visual resonance of "Hive-Mind" sound sculpture: clay bowls as driven coupled oscillators

"Hivemind" explores the sonic potentials of ceramics by revealing the pitched resonance of porcelain bowls using a coupled-oscillator mechanical system. Two reciprocating platforms are populated with over 300 clay vessels with marbles placed about the inner bowls. By modulating the speed of the applied pushing motion (see Fig. 5), this piece surveys the acoustic potential of ceramic as material by exposing the audio-visual "resonance" of different bowls. When this pushing motion matches the natural rotational frequency of the bowl's topography, the marble begins to rotate and loop with more velocity thereby amplifying the characteristic resonance of the bowl. Because each bowl contains a different resonant frequency, clusters of similarly-sized bowls can be amplified to create slowly-changing bell-like sonorities. The pushing motion of the two platforms drive the system into different dynamic states to form a time-based composition of audio-visual resonance.

<sup>1</sup> for more information, see my code repository: [https://bitbucket.org/no\\_lem/kura-python/](https://bitbucket.org/no_lem/kura-python/)

From the perspective of coupled oscillation, the reciprocating platform can be thought of as a type time-dependent external forcing factor,  $\Lambda_e(\phi_i, t)$  from Equation (5) that drives the system. Because the marbles' motion are beholden to this external force, each bowl can be looked at as a resonant filter at some center frequency determined by their shape. The input to these filters is simply the pushing motion by the reciprocating platform and their audible output is the sum of their (damped) oscillations. Even though the individual marbles are not explicitly coupled to one another, they resonate in concert with the frequency (and amplitude) of the external sinusoidal pushing force. To explicitly couple the oscillators, one would have to resort to a different physical implementation that would allow the instantaneous phase of each physical object to interact with the others.

## 6. CONCLUSIONS

This paper looked at the extent to which coupled oscillators can be useful to describe a wide range of musical phenomena by demonstrating several ways in which they model synchronous auditory phenomena. There's still much territory to be explored in this area of applied musical research. For instance, this paper only looked at one such synchronization scheme—the Kuramoto model—to describe a type of self-organization. There are several other synchronization models (pulse-coupling, sync and swarm models, Van Der Poll oscillators, etc.) that could be exploited in the context of art and music generation. Similarly, this paper only briefly mentioned several applications related to digital signal processing, rhythmic generation, or music perception. One particularly promising area of research is neural resonance theory in the context of beat induction and meter perception as posed by Large [2]. As an outgrowth of dynamic attending theory, his canonical model accounts for the entrainment of endogenous cortical rhythms from the acoustic rhythms of the external world [19]. More importantly, his canonical model is derived from a coupled-oscillator model of dynamical systems. Future research learning how to integrate these notions of perceived beat and rhythm into different generative models would be well served in the area of music creation and sound art.

## 7. REFERENCES

- [1] A. Ravignani, D. Bowling, and W. T. Fitch, "Chorus-ing, synchrony and the evolutionary functions of rhythm," *Frontiers in Psychology*, vol. 5, no. SEP, pp. 1–15, 2014.
- [2] E. Large, "Neurodynamics of Music Perception," Stanford, CA, 2014. [Online]. Available: <https://www.youtube.com/watch?v=mPaR9PqgPXc>
- [3] T. Blackwell, "Swarm Music: Improvised Music with Multi-Swarms," *The 2003 AISB Symposium on Arti-*

- icial intelligence and Creativity in Arts and Science, pp. 41–49, 2003.
- [4] C. Roads, *Microsound*, pap/cdr ed ed. MIT Press, 2004.
- [5] J. Smith, *Physical Audio Signal Processing: for Virtual Musical Instruments and Digital Audio Effects*. W3K Publishing, 2010.
- [6] A. Lambert, “a Stigmergic Model for Oscillator Synchronisation and Its Application in Music Systems,” *Proceedings of the International Computer Music Conference*, pp. 247–252, 2012.
- [7] S. Strogatz, “From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators Steven,” *Physica D*, vol. 143, pp. 1–20, 2000.
- [8] S. H. Strogatz, “Exploring complex networks,” *Nature*, vol. 410, no. March, pp. 268–276, 2001.
- [9] Y. Kuramoto, “Self-entrainment of a population of coupled non-linear oscillators,” *International Symposium on Mathematical Problems in Theoretical Physics, Lecture notes in Physics*, vol. 39, no. H, pp. 420–422, 1975.
- [10] J. Snyder, A. Zlotnik, and A. Hagberg, “Stability of entrainment of a continuum of coupled oscillators,” *Chaos*, vol. 27, no. 10, pp. 1–25, 2017.
- [11] M. J. Richardson, R. L. Garcia, T. D. Frank, M. Gergor, and K. L. Marsh, “Measuring group synchrony: A cluster-phase method for analyzing multivariate movement time-series,” *Frontiers in Physiology*, vol. 3 OCT, no. October, pp. 1–10, 2012.
- [12] G. Ligeti, “Poème Symphonique,” 1962.
- [13] S. Boda, Z. Nédá, B. Tyukodi, and A. Tunyagi, “The rhythm of coupled metronomes,” *European Physical Journal B*, vol. 86, no. 6, 2013.
- [14] A. Neda, Z. Ravasz, E. Vicsek, T. Brechet, Y. Barabasi, “Physics of the rhythmic applause,” *Theoretical and Mathematical Physics*, vol. 104, no. 3, pp. 1104–1107, 1995.
- [15] E. Buck, John; Buck, “Synchronous Fireflies,” *Scientific American*, vol. 234, no. 5, pp. 74–85, 1976. [Online]. Available: <https://www.jstor.org/stable/24968881>
- [16] A. Ravignani, D. Bowling, and S. Kirby, “The Psychology of Biological Clocks: A New Framework for the Evolution of Rhythm,” *The evolution of language: Proceedings of the 10th International Conference*, no. September, pp. 270–277, 2014.
- [17] W. Mertens, *American Minimal Music: La Monte Young, Terry Riley, Steve Reich, Philip Glass*, A. Broude, Ed. Kahn & Averill, 1983.
- [18] A. Licht, “Organizing the Unpredictable | Rhizome,” *Rhizome*, 2009. [Online]. Available: <http://rhizome.org/editorial/2009/feb/18/organizing-the-unpredictable/>
- [19] E. W. Large, F. V. Almonte, and M. J. Velasco, “A canonical model for gradient frequency neural networks,” *Physica D: Nonlinear Phenomena*, vol. 239, no. 12, pp. 905–911, 2010. [Online]. Available: <http://dx.doi.org/10.1016/j.physd.2009.11.015>

# “JAZZ MAPPING” AN ANALYTICAL AND COMPUTATIONAL APPROACH TO JAZZ IMPROVISATION

**Dimitrios Vassilakis**  
Department of Music Studies  
National and Kapodistrian  
University Of Athens  
info@dimitriosvassilakis.com

**Anastasia Georgaki**  
Department of Music Studies  
National and Kapodistrian  
University of Athens  
georgaki@music.uoa.gr

**Christina Anagnostopoulou**  
Department of Music Studies  
National and Kapodistrian  
University Of Athens  
chrisa@music.uoa.gr

## ABSTRACT

“Jazz mapping” is a multi-layered analytical approach to jazz improvisation. It is based on hierarchical segmentation and categorization of segments, or constituents, according to their function in the overall improvisation. The approach aims at identifying higher-level semantics of transcribed and recorded jazz solos. At these initial stages, analytical decisions are rather exploratory and rely on the input of one of the authors and experienced jazz performer. We apply the method to two well-known solos, by Sonny Rollins and Charlie Parker, and discuss how improvisations resemble story-telling, employing a broad range of structural, expressive and technical tools, usually associated with linguistic production, experience, and meaning. We elucidate the implicit choices of experienced jazz improvisers, who have developed a strong command over the language and can communicate expressive intent, elicit emotional responses, and unfold musical “stories” that are memorable and enjoyable to fellow musicians and listeners. We also comment on potential artificial intelligence applications of this work to music research and performance.

## 1. INTRODUCTION

**1.1 Goals:** The project aims at advancing our current understanding of jazz improvisation and, by extension, of musical creativity. It introduces and applies a musical language-mapping scheme that can support the creation of a large annotated corpus of transcribed solos, assist in the pedagogy of improvisation and serve as a reference point in human and artificial musicianship research. The utility of the approach may also extend to research in other domains that explore hierarchical sequential data and real-time decision making, such as generative modeling of natural language and speech.

**1.2 Related work:** Formal music analysis is usually concerned with breaking the musical surface into segments based on similarity, and with studying how these are put together syntactically as a piece of music unfolds in time, thus attributing internal cohesion [1] Semiotic analysis (paradigmatic and syntagmatic) is a typical example of a method which categorizes segments according to similarity [2] Paradigmatic analysis has been computationally modeled in the past [3], [4].

At the same time, a significant body of research literature addressing jazz improvisation has been developing over the last couple of decades. This work includes topics on jazz storytelling [5], including references on the concept by well known jazz musicians and scholars. Some [6] introduce the concept of re-telling to refer to the re-working of a standard, based on a famous recording of a master, stressing the important tension between individual voice and tradition. Others [7] explore machine learning of jazz grammars, using basic building-blocks or “slopes,” touching upon the antitheses of abstraction versus vocabulary, and attempting to codify harmonic tension.

A relevant work that focused on Sonny Rollins’s thematic improvisation [8] will be explored further, below.

Weimar’s Jazzomat Research Project [9] has produced several databases of annotated solos and licks, including the “Dig That Lick” database. Studies on the use of patterns in jazz [10], [11], have stressed the importance of auditory and motor patterns organizing into a stored menu of pattern libraries.

Researchers at the Georgia Institute of Technology have been developing robotic applications of computer improvisation [12] that are informing and are being informed by our work.

Francois Pachet in 2001 produced The Continuator, later developed in the European project MIROR (mirrorproject.eu) [13], focuses on learning sequences by linear analyses of input patterns and phrases to generate a response. Improvisations have been generated in real time based on input of musical sequences [14]. Explorations on the improvisers’ thought processes during a duo [15] have attempted to reveal the intent and the scheme or scenario behind an improvisation. Musical passage coding as “phrase” and “variation” has been used to assist a music program to acquire “common sense,” [16], while a very interesting interview of Ornette Coleman by Jacques Derrida touches on the relationship between language and jazz improvisation.<sup>1</sup>

All the above approaches deal with a structural analysis of jazz improvisations, thus studying the jazz vocabulary

<sup>1</sup>[http://www.ubu.com/papers/Derrida-Interviews-Coleman\\_1997.pdf](http://www.ubu.com/papers/Derrida-Interviews-Coleman_1997.pdf)  
Interview originally appeared in French in the magazine Les Inrockuptibles no. 115 (20 aout-2 septembre 1997): 37-40,43.

and syntax, but they are not progressing deeper into the semantics of the language.

Based on the above approaches, and while we acknowledge that the topic of semantics in jazz might be too complex to describe with a formal syntactic analysis, we make a first attempt in interpreting the various constituents that result from the analysis, together with their function and style in the improvisation, expanding into issues of semantics, syntactical analyses, story telling and thematic development.

**1.3 Proposed outcome:** The “jazz mapping” project has potential implications to machine learning and Artificial Intelligence (AI) system development. It can provide means for AI to manage in a human-like way the essential human tension among past, present, and future characterizing all decision-making. This potential can be realized through “teaching” an AI system the rules that govern annotation and how these rules dynamically interact and change when actualized as experienced present or “now”.

We will begin by identifying and adapting to jazz improvisation musical contexts basic human communication tools/codes, concepts and structures such as: question and answer/call and response, fragment, lick, phrase, thematic development, short/long, memorable or abstract, and references among phrases. A similar approach can potentially be used to explore concepts such as harmonic tension, phrasing, articulation, expressiveness, sonic character or “sound,” *etc.*, to generate jazz solos much like a jazz improviser/storyteller would, using layers of multi-reference.

A pattern database will be also created as those annotated phrases licks, fragments and patterns will have multiple uses on “describing” or “outlining” chords and chord sequences helping to address issues like originality and personal voice and different approaches of players like vertical versus linear, voice leading versus modal or free.

## 2. THE JAZZ MAPPING APPROACH

### 2.1 Constituents in syntactic analysis.

In order to analyze an improvisation through mapping we propose a novel method which consists of the following levels: Jazz improvisational structural elements and mappings, thematic analyses by defining segments, licks and phrases and annotation of syntax and structure.

In our analysis, we define 3 types of constituents, listed here by increasing duration and/or complexity:

1. **Segment**
2. **Lick**
3. **Phrase**

Each of the constituents found would carry a tag describing the function in the improvisation, such as: *response/answer*, *reference*, or *new idea*.

### 2.2 Definitions

Below we attempt a definition for each constituent, bearing

in mind that this is not a fully formal approach yet, therefore the criteria for a constituent to belong to a category are not fully explicit, and rely to some extent on the context of the piece under analysis.

**Segment:** very short but salient theme, fragment, angular/linear/long single note, usually one bar (*e.g.* the thematic seed in John Coltrane’s “Love Supreme” Ex.1).



Ex.1

(John Coltrane goes on to build part of his solo using this fragment in different keys).

Segment duration does also depend on tempo; visual analogue: a Lego piece or a brick.

**Lick:** a memorable theme usually between two and four bars (*e.g.* Ex.2, the opening in Charlie Parker’s “Now’s The Time”).



Ex.2

Lick is longer than a segment and shorter than a phrase (again, dependent on tempo, typically not longer than four bars); often musicians transpose favorite licks in a variety of keys to enhance their “vocabulary” in a certain style; can also be used as “mannerisms” to reference another performer or style; visual analogue: a larger, more salient and recognizable structure such as a door or a window.

**Phrase:** longer sequence of notes<sup>2</sup> that may or may not contain discernible segments or licks; visual analogue: an entire room or part of a space that can contain legos/bricks, doors, windows, *etc.*

Here is Ex.3, Dexter Gordon’s 7 bar long phrase from “Cheesecake”.



Ex.3

Additionally a constituent, according to its function, would acquire one of the following characterisations: a **response/answer** to a previous element in the same piece (reaction to an internal/local musical event), a **reference** to a previous element in the same piece (allu-

<sup>2</sup> Our initial focus on horn solos imposes a maximum phrase duration based on breath capacity, which can, of course, be exceeded when using circular breathing techniques.



sion to an internal/local event), or an independent new idea. In terms of score annotation for the mapping we use **S** for segment, **L** for lick, **P** for phrase, different colours for each one, plus **r** for reference/relationship, **a** for answer/response, while location is described with brackets.

### 3. ANNOTATION SYNTAX

In annotating the above constituents as a piece unfolds in time, i.e. syntactically, we developed textual annotations to describe constituents and their locations.

#### 3.1 Location

Location is annotated as “measure number and beat number within the measure”. For example, location “1.3” means” third beat of the first measure”. Longer durations are annotated analogously. For example, an element’s duration of “1.1 - 2.4” means that the element starts on beat 1 of measure 1 and ends on beat 4 of measure 2.

#### 3.2 Constituents and Qualifiers

Segment = **S**(Index, Reference, Response)

Lick = **L**(Index, Reference, Response)

Phrase = **P**(Index, Reference, Response)

**Index:** numerical order of appearance of a structural element

**Reference:** 1,2,3... = a first/second/third reference; 0 = not a reference

**Response:** 1,2,3... = a first/second/third response; 0 = not a response

If both **Reference** and **Response** are 0 the element qualifies as a **New Idea**.

#### 3.3 Annotation Example

For Measure 1 in Sonny Rollins’s “St. Thomas” we would write **1.1 - 1.4; S(1, 0, 0)** to indicate: beats 1-4 of measure 1 outline the first distinct segment of the piece which is not a reference or response to any other element but a new idea.

For Measure 2 we would write **2.1 - 2.4; S(1, 0, 1)** to indicate: beats 1-4 of measure 2 constitute the 1st response (and not a reference) to the 1st segment, which was introduced in measure 1.

For Measures 15-17 we would write **15.1 - 17.1; S(1, 2, 3)** to indicate: the portion beginning at measure 15, beat 1 and ending at measure 17, beat 1 constitutes the 2nd reference and 3rd response to the 1st segment.

#### 3.4 Additional definitions

##### \*: Mannerism

A segment, lick, or phrase that exemplifies a performer’s style; a structural element that sounds like a quintessential Sonny Rollins, Charlie Parker, or any artist passage.

For example: **S(1,0,0)\*** describes the piece’s 1st segment, which is neither a reference nor a response, nor a wholly new idea but, rather, a stylistic mannerism, pointing to a specific style characteristic to an artist or genre.

This designation helps differentiate between references to elements within a given solo and references to the performing artist’s “memory bank.”

The following Ex.4 is a Parker mannerism on “Now’s The Time”, that we see in a more elaborate version below at our analyses of “Au Privave”.



Ex.4

##### \*\*: Quote

A segment that directly incorporates a well-known and recognizable structural element from another piece (e.g. a theme from Beethoven’s 5th symphony, a lick from a Jazz standard or a well-known pop song, or another player’s favorite phrase).

The use of quotes in jazz improvisation is happening often so if the quotes are properly labelled inside a well-formed database of phrases, fragments and licks, then we can annotate adding specifically the source of the quote and a double asterisk: **S(1,0,0)\*\***.

## 4. ANNOTATION EXAMPLES

#### 4.1 Sonny Rollins solo on “St. Thomas”.

Score analyses with brackets and annotation definitions. (We also use colors to help identify the constituents Segment=green, Lick=red, Phrase=blue):

Sonny Rollins

St. Thomas

1.1 - 1.4; S(1, 0, 0)  
2.1 - 2.4; S(1, 0, 1)  
3.1 - 3.4; S(1, 0, 2)  
4.1 - 5.1; S(1, 0, 3)  
5.2 - 5.4; S(1, 1, 0)  
6.1 - 6.4; S(1, 1, 1)  
7.1 - 7.2; S(1, 1, 0)  
7.3 - 8.1; S(1, 1, 2)  
8.2 - 9.1; S(1, 1, 3)  
9.3 - 13.2; L(1, 0, 0)  
13.1 - 13.3; S(1, 2, 0)  
13.3 - 13.4; S(1, 2, 1)  
14.2 - 14.4; S(1, 2, 2)  
15.1 - 17.1; S(1, 2, 3)  
17.1 - 17.4; S(1, 0, 0)  
17.4 - 18.2; S(1, 0, 1)  
18.2 - 18.4; S(1, 0, 2)  
19.1 - 19.4; S(1, 0, 3)  
20.1 - 21.2; S(1, 0, 4)  
21.3 - 24.4; L(2, 0, 0)

Annotation:

1.1 - 3.4; L(1, 0, 0)  
 4.4 - 5.3; S(1, 0, 0)  
 5.4 - 6.3; S(1, 0, 1)  
 7.1 - 11.4; P(1, 0, 0)  
 12.4 - 19.4; P(2, 0, 0)  
 20.1 - 23.1; L(2, 0, 0)\*  
 23.2 - 24.1; S(2, 0, 0)  
 24.2 - 25.1; S(2, 0, 1)  
 25.2 - 27.4; L(3, 0, 0)  
 28.1 - 29.4; L(4, 0, 0)  
 30.1 - 31.1; L(4, 0, 1)  
 31.4 - 33.1; L(5, 0, 0)  
 33.3 - 35.4; L(6, 0, 0)  
 36.1 - 37.4; L(7, 0, 0)

### 4.3 Comments on the 2 solos

For this paper we analyzed 2 solos from different periods of jazz and from different players. We see a much longer solo on Sonny Rollins, as it is later hard bop period, and he is thus able to expand into thematic development, while Charlie Parker takes a much shorter solo on the blues but he is the one who presented the new bebop language that forms the basis of modern jazz improvisation to this day. He doesn't refer back to himself like Sonny was able to do later on, he introduces new ideas and also plays one of his favorite phrases on the double time that since then has become a sort of parkerism for the jazz community. We have a sense that Parker was able to play so much "music" in a very short solo, while Sonny on a longer solo creates movement, interest and innovation by his thematic development approach.

We see how Sonny Rollins uses the opening segment to built thematic development in many instances of the solo, not only as related segments, but also as part of licks and longer phrases. These elements mark a great development in the syntax and the story telling of a jazz solo.

Both players share the love of the blues, a very basic element in jazz improvisation and their both have a great swing "feel".

Many of the above segments, licks and phrases are part of the jazz vocabulary of today and we witness here the development of jazz from two masters of their art, who among others defined the language and also created a very strong personal voice.

## 5. METHODOLOGY DISCUSSION

### 5.1 Sequential information (Thematic development)

Identifying locations in time for each element provides the structural skeleton that can support future automation of such analyses and AI-system-generated thematic development.

For example a sequence may proceed as:

*Segment, answer, answer, lick, related segment, answer, repeat, original segment, new lick, new segment, answer,*

*phrase(that may or may not contain previously introduced segments or licks), first lick reference, answer etc.*

Codification of sequential development may also find applications in speech analysis and several temporal art forms.

### 5.2 How to call and answer

There are plenty of instances of this paradigm in improvisation. What we would learn is the transformation function that takes us from the initial structural element (such as the segment, lick, or phrase) into the response. Similarity or contrast can both form the basis of a question/answer procedure.

We also have information that describes the sequence of the responses so we could learn how the first response differs from the second, or the third, and so on.

For example, in the first 4 measures of Sonny Rollins on St. Thomas we see there is an initial segment, a response segment, a 2nd response segment, and a third response segment. In this example each response has more notes than the previous. Such trends are learnable.

### 5.3 Transformations or referencing and embellishing

This has similarities to the call / response paradigm. However, a reference to a previous element is not necessarily a "response" but can serve a different thematic structure function.

Repeated phrases: here we either annotate as the same segment/lick/phrase, but in the case of small alterations to the original then this again is mapped as reference and answer.

### 5.4 Hierarchical

Three examples to look out for:

- a) Combine segments to create licks
- b) Combine licks to create phrases
- c) Freely combine all three elements

While there are instances where a lick or phrase is made of smaller elements, not every lick or phrase can be described this way. Often, licks and phrases are original and do not reference other elements.

### 5.5 Structural interchange

Cross-reference among the three identified structural elements provides another means of thematic development during improvisation. Our analytical approach can capture this feature through double annotation on the specific bar or bars. See, for example, the end of Lick1 and Phrase1 on the Sonny Rollins solo where he ends restating the 1<sup>st</sup> segment idea.

### 5.6 Voice leading concept

In be-bop, hard-bop and modern jazz styles voice leading is frequently used to end or connect licks, phrases and themes. In a more open, modal or free playing this is not so evident. Rather, harmonic tension, sound, articulation

and note density within phrases provide the most important cues. We anticipate that an upcoming multi-layered mapping will address this issue.

### 5.7 Emotion and creativity

Emotion: A common mechanism in music, also employed here, is creating patterns of tension and release that play with the listeners' expectations.

To what extent something can be characterized as interesting or emotional is contingent on what preceded it and what, eventually, follows. A player known for a specific style or mannerism – say, a linear approach – can inhibit expectations by switching to a vertical approach, or by inserting unexpected pauses, long notes, or sound effects. Variations such as these that increase interest and elicit affective responses are manifestations of the performer's creativity and capacity to unfold a musical improvisation as a compelling story.

### 5.8 Inspiration

One way to approach “inspiration” could be in terms of compelling, unexpected structures that arise out of randomness. As jazz musicians deal with randomness, if suddenly - in playing or practice - we get a structure/phrase that stands out in terms of being memorable or highly organized/structured then we recognize this as inspiration that usually becomes a new composition or a favorite mannerism.

### 5.9 Thematic development and multi reference

References to previous elements, whether as straight repeats or augmented, diminished, displaced, or otherwise modified, can be considered a form of self-reference. Feeding a database of such manipulations and thematic developments to machine learning algorithms can support the development of AI systems that exhibit self-referential behavior and, by extension, apparent self-awareness.

## 6. CONCLUSIONS

We have proposed an analytical method that supports systematic annotation of a wide variety of jazz solos and can reveal the musical language characteristics of individual players and styles. The annotated constituents per solo will eventually feed a database of musical segments, licks, and phrases that can imply and outline a specific chord or a longer harmonic progression. We anticipate that this database will enhance the “bag of tricks” of the jazz player and help the jazz educator explain jazz styles, performers' personal voices, and characteristic mannerisms.

In jazz, performers always strive to develop a personal voice that can stand next to that of the masters. The knowledgeable player or listener can usually identify, after only a few notes, a master performer who has developed language and mannerisms that are immediately evident.

A personal voice consists of sounds and sound structures with certain recognizable and personal qualities that function as a performer's signature. The mappings supported in this study can help reveal and codify these signatures and organize them into systematic categories.

Further work is required to better define stylistic constituents, flexible enough to codify a broad range of styles and personal voices. As we proceed, we will seek the insights of top jazz improvisers, worldwide, and assess the resulting database through AI machine learning and performance.

### Acknowledgments

We would like to thank Gil Weinberg, Alexander Lerch, Mason Bretan, Frank Clark, and Athanassios Economou (Georgia Institute of Technology), Martin Norgaard and Mariana Montiel (Georgia State University) and Pantelis Vassilakis (AcousticsLab) for their advice, friendship, wisdom and research collaborations.

Deep thanks to David Liebman and fellow players Jeff “Tain” Watts, Essiet Okon Essiet, Sylvia Cuenca, Benito Gonzales, Ralph Peterson, Dave Kikoski, Theo Hill, Craig Bailey - among many others - that helped by their playing, discussions and insights.

We would also like to thank the graduate students at the Department of Music Studies, National and Kapodistrian University of Athens.

### REFERENCES

- [1] N. Cook, *A Guide to Musical Analysis*, OUP, 1987.
- [2] J.J. Nattiez, *Fondements d'une Semiologie de la Musique*, Union Generale d' Editions, 1975.
- [3] C. Anagnostopoulou and G. Westermann, “Categorisation in music: A computational model for Paradigmatic analysis,” *Proceedings ICMC*, Thessaloniki, Greece, 1997.
- [4] E. Cambouropoulos, *Towards a general computational theory of musical structure*, PhD Thesis, University of Edinburgh, 1998.
- [5] F. Okiji, “Storytelling in Jazz Work as Retrospective,” *Collaboration Journal of the Society for American Music*, 11(1), February 2017, pp. 70-92.
- [6] S. Bjerstedt, *Story Telling in jazz Improvisation*, PhD Thesis, Lund University, 2014.
- [7] J. Gillick, K. Tang, and R.M. Keller, “Machine Learning of Jazz Grammars,” *Computer Music Journal*, 2008, pp. 111–222.
- [8] G. Schuller, “Sonny Rollins and the challenge of thematic improvisation,” *The Jazz Review*, 1(1). 1958.
- [9] J. Abeßer, E. Cano, K. Frieler, and M. Pfeleiderer, “Dynamics in jazz improvisation – score-informed estimation and contextual analysis of tone intensities in trumpet and saxophone solos,” *Proceedings of the*



*9th Conference on Interdisciplinary Musicology – CIM14*, Berlin, Germany, 2014.

- [10] M. Norgaard, M. Montiel, and J. Spencer, “Chords not required: Incorporating horizontal and vertical aspects independently in a computer improvisation algorithm,” *Proceedings of the International Symposium on Performance Science*, 2013, pp. 725–730.
- [11] M. Norgaard, “How jazz musicians improvise: The central role of auditory and motor patterns,” *Music Perception*. 31(3), 2014, pp. 271-287.
- [12] M. Bretan and G. Weinberg, “A survey of robotic musicianship,” *Communications of the ACM*, 59(5), 2016, 100-109.
- [13] A.R. Addressi, L. Ferrari, S. Carlotti, and F. Pachet, “Young children musical experience with a flow machine,” *Proceedings of ICMPC9 and 6th ESCOM Conference*, 2006.
- [14] G. Assayag and S. Dubnov, “Using Factor Oracles for machine Improvisation,” *Soft Computing*, 8(9), 2004 c; Imperial College Press, Lecture Notes Series, 2016, pp.61-74.
- [15] D. Mendonca and W. Wallace, “Cognition in Jazz Improvisation: An Exploratory Study,” *Proceedings of the Annual Meeting of the Cognitive Science Society*, 26(26) 2004.
- [16] D. Horowitz, “Representing Musical Knowledge in a Jazz Improvisation System,” MIT Media Lab, Cambridge, MA, 1995.

# Visual Pitch Estimation

**A. Sophia Koepke**

University of Oxford

koepke@robots.ox.ac.uk

**Olivia Wiles**

University of Oxford

ow@robots.ox.ac.uk

**Andrew Zisserman**

University of Oxford

az@robots.ox.ac.uk

## ABSTRACT

In this work, we propose the task of automatically estimating pitch (fundamental frequency) from video frames of violin playing using *vision* alone. Here, we consider only monophonic violin playing (where only one note is being played at a time).

In order to investigate this task, we curate a new dataset of monophonic violin playing. We propose a Convolutional Neural Network (CNN) architecture that is trained using a student-teacher strategy to distil knowledge from the audio domain to the visual domain. At test time, our network takes video frames as input and directly regresses the pitch. We train and test this architecture on different subsets of our new dataset.

We show that this task (i.e. pitch prediction from vision) is actually possible. Furthermore, we verify that the network has indeed learnt to focus on salient parts of the image, e.g. the left hand of the violin player is used as a visual cue to estimate pitch.

## 1. INTRODUCTION

Humans can obtain some understanding of music simply by watching instruments being played, even without access to audio recordings of the music itself. Indeed, a trained musician might be able to transcribe an entire video purely from visual cues alone, although with great painstaking manual effort. The movement and position of the instrument and body (specifically the movement of the arms, hands and fingers) have a direct correlation with the sound produced. In this work, we investigate the following question: is it possible for a trained neural network to identify the pitch of played notes, simply from the frames of a silent video?

Our approach is a valuable first step towards the task of *complete* visual music transcription. While *audio* based music transcription is a widely studied and successful field, the task of *visual* music transcription has not been explored to a great extent. Performing this task from standard frame-rate visual information alone can be extremely useful in instances when the audio is of poor quality, missing, or mixed with information from other audio sources, e.g. in

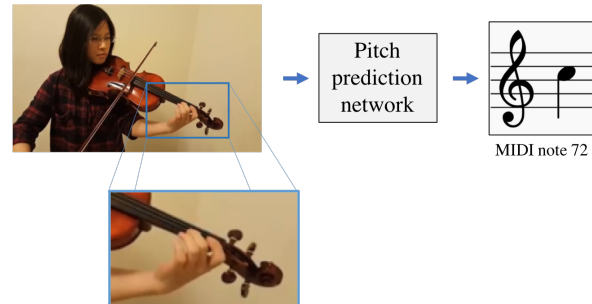


Figure 1. Pitch estimation from visual information. Given video frames, the network is tasked to predict pitch using *only* the visual information.

the case of polyphonic music. These scenarios are challenging for purely audio-based pitch estimation methods.

We investigate this by training a network to predict pitch information from video frames of monophonic solo violin recordings using only the visual image data (see Figure 1). Given a set of video frames, the network learns to regress the corresponding pitch. In order to perform this challenging task, our method makes use of two insights. First, using a teacher-student strategy (i.e. training one network using another network [1]) is important to enforce that the network learns the visual cues that are correlated with the corresponding sound. Second, using multiple frames as input (i.e. a short silent video clip) is preferable to using a single still frame. This is because the additional frames resolve ambiguities such as which string is vibrating (i.e. the string that is being played on with the bow). These insights inform our architecture choices, described in Section 3.

The models are trained and evaluated on a new dataset (Section 4) of violin playing. This dataset is divided into three subsets which vary in difficulty. The first two subsets are recordings of a single player photographed by a fixed mobile phone camera. The third subset consists of ‘in-the-wild’ videos downloaded from YouTube.

On all of these datasets, our method demonstrates that regressing pitch directly from video frames is indeed possible (Section 5). Finally, we verify that the method is making sensible predictions by investigating what regions of the image are most salient for the prediction. We find that our method focusses on the movement and location of the musician’s arms, hands and fingers; this is similar to how a human would approach this task.

Copyright: © 2019 A. Sophia Koepke et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

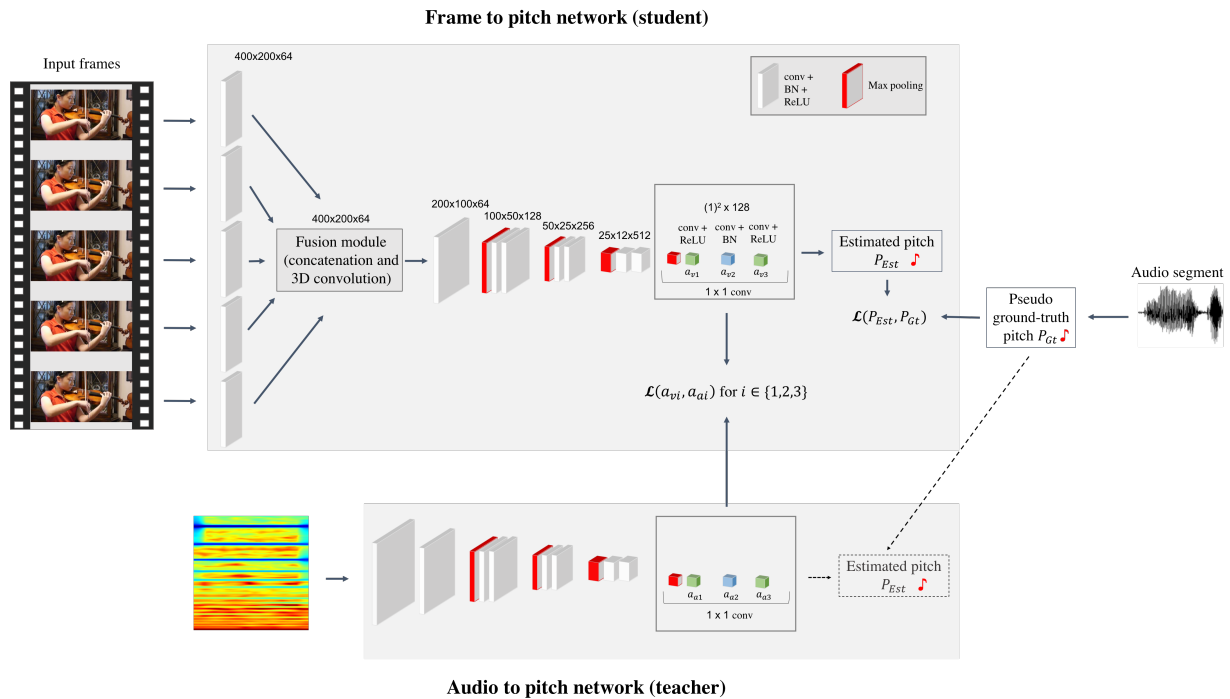


Figure 2. An overview of the visual pitch estimation method. We train our framework with a student-teacher strategy by distilling discriminative knowledge from a teacher network to a student network. The teacher network regresses pitch from audio whereas the student network is trained to regress pitch from visual information alone. Both networks are trained using pseudo ground-truth pitch information which is automatically extracted using an audio-based method and thereby does not require any manual annotation. The audio to pitch network is trained first and then used to train the frame to pitch network (student) by minimising the distance between the activations of the final three fully-connected network layers of the student and the teacher network. At test time, given one or multiple visual input frames, the student network is used by itself to regress pitch. In the case of multiple input frames, the outputs of the first convolutional layers of the student network are concatenated and fused through a 3D convolutional layer (Fusion module) before being input to the next convolutional layer.

## 2. RELATED WORK

Here, we only consider directly related work on cross-modal information transfer between audio and visual information.

**Multi-modal audio-visual representations.** Training strategies that encourage synchronisation between the audio and visual streams have been used successfully for speech synchronisation [2]. More generally the correspondence between the audio and visual streams (though not their strict synchronisation) has proven very useful for obtaining meaningful features for sound localisation and separation [3,4]. In these works, the natural synchronisation in videos can be leveraged in a self-supervised manner to obtain useful image and audio representations. Aytar et al. [5] also exploit this natural synchronisation property in order to transfer knowledge from visual recognition networks into sound networks. However, we propose a framework that transfers knowledge the other way round, i.e. from audio to visual information.

**Cross-modal audio-visual generation.** Related to our framework are methods that generate audio from visual information, e.g. spectrograms or other sound features from visual information [6–10], or localise sound in video in order to separate different sounds [11, 12], or analyse vibrato-

patterns for audio-visual association [13].

The URMP dataset by Li et al. [14] is targeted at cross-modal audio-visual generation. However, it poses two limitations for our task. Firstly, the image resolution of the released dataset is not very high which makes it difficult to actually recognise pitch from the visual information alone, as the key parts of the image (e.g. the fingers of the left hand) are too small. Secondly, the dataset was recorded in constrained settings with only a limited number of musicians which limits the generalisability of models trained on this data to other settings. Therefore, in addition to training and evaluating our models on the URMP dataset, we gathered a new dataset to train and test our framework that is of higher resolution and which also contains ‘in-the-wild’ videos.

**Music transcription from silent video.** More closely related to ours is the work by Gomez et al. [15] which proposes to leverage visual information to transcribe clarinet videos using the hand movements in recorded video sequences. However, unlike their method, we do not require any manual tracking or labelling (i.e. finger/hole positions) as supervision in order to train our network.

Zhang et al. [16] addressed a similar task to ours of visu-

ally obtaining pitch for violin by detecting the strings of a violin and by recognising finger events (such as their position and whether they are pressing on a string). However, their method is quite constrained; it involves tracking the fingers and the strings, and makes assumptions about the length of the fingerboard which requires the image data to always be perfectly aligned. In contrast, our method gives convincing results for different viewpoints and requires no manual labelling.

Another related method is the physics-based approach for recovering pitch from silent guitar video by Goldstein and Moses [17]. However, their method requires mounting a camera, that allows recording with high frame rates, on the guitar itself in order to use the actual string vibrations to predict pitch. Unlike their method, our set-up only requires the use of a normal camera and it learns to localise the left-hand position of the musician (relative to the instrument) in order to infer pitch. Our method can thus be applied retroactively to videos that have already been recorded.

### 3. MODEL

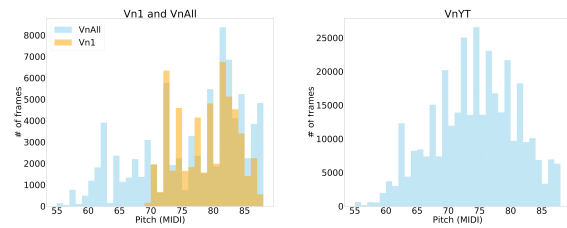
In this section, we describe the training and testing framework used to regress pitch from video frames. We treat this as a classification task. The network takes video frames as input and estimates the pitch as a MIDI number. An overview is given in Figure 2.

**Teacher-student strategy.** We found that directly regressing pitch from the video frames did not generalise at test time. This is presumably because the visual information relevant for the pitch prediction task occupies only a small part in the video frames.

As a result, we train two networks – a teacher and student network – such that the activations of the student network are similar to those of the teacher. The teacher network regresses pitch from audio and the student regresses pitch from video frames. The rationale for using this strategy is that, in order to contain relevant information about pitch, the high-level representation of the visual information (encoded in the student) should be close to that of the audio information (encoded in the teacher). This strategy proved crucial to obtain a network that generalises at test time.

The teacher network is first trained using STFT spectrograms as input to regress the pseudo ground-truth pitch (the method for obtain this pseudo ground-truth is described in section 5). The student network is then trained to regress the pitch with an additional loss that enforces that the activations of the higher level layers are similar to those in the teacher network. For this, we use an L1 loss which is weighed so that the contribution for each of the three fully-connected layers is as big as the pitch classification loss. Both networks are trained to predict pitch with a cross-entropy loss.

**Neural Network Architecture.** The teacher and student network architectures are loosely based on the VGG-M network architecture [18] and can be seen in more detail in Figure 2. For the student network, in the case of multiple input frames, the outputs of the first convolutional layers are concatenated and fused using a 3D convolutional layer to combine the information from the frames with spa-



(a) *VnI* and *VnAll* datasets.

(b) *VnYT* dataset.

Figure 3. Pitch distribution over the number of frames in the three subsets of our dataset; the constrained single-string data *VnI*, the data on all strings *VnAll*, and the in-the-wild data *VnYT*. Pitch is shown in MIDI numbers. The subsets cover the chosen full pitch ranges.

tial kernel size  $3 \times 3$  followed by batch normalization and ReLU. The output serves as input to the second convolutional layer. For just a single input frame, the output of the first (2D) convolutional layer is directly input to the second convolutional layer. The first two convolutional layers consist of  $7 \times 7$  convolutions, whereas the subsequent ones are  $3 \times 3$  convolutions and the last three are  $1 \times 1$  convolutions. All convolutional layers have stride 1 except for the second one, which has stride 2.

### 4. DATASETS

We curate a new violin playing dataset which consists of three subsets (*VnI*, *VnAll* and *VnYT*) that differ in difficulty and size. The most challenging subset *VnYT* consists of in-the-wild violin solo videos downloaded from YouTube<sup>1</sup>. These are largely comprised of recordings of solo recitals, etudes, and orchestra auditions. Both *VnI* and *VnAll* are recorded in simpler conditions: all videos are of a single violinist, have similar backgrounds and are taken from similar angles.

The datasets vary in terms of the range of pitches. Both *VnI* and *VnAll* consist of recordings of a violinist playing in a practise-like set-up (but without the thousandfold repetitions of the same phrases). The easiest subset *VnI* consists of videos that only contain violin playing on a single string, resulting in a range of 20 semitones (MIDI numbers between 68 and 88). Estimating the pitch is easier in this case, as there is no ambiguity concerning which string is being played. *VnAll* contains videos played in the full pitch range of the violin without being restricted to playing just on one string. For both *VnAll* and the most difficult subset *VnYT*, we consider a range of 33 semitones (MIDI numbers between 55 and 88).

All subsets are split into train/val/test sets. Disjoint parts of the same videos are used for training and validation. The test sets consist of frames that were not seen during training (from left-out unseen videos). The precise numbers of frames and videos are given in Table 1.

<sup>1</sup> Example videos: <https://youtu.be/-ccYdhQAn10>, <https://youtu.be/YGCYelAHdaU>



Dataset <i>VnI</i>			
	Train	Val	Test
# of videos	5		1
# of frames	25308	2791	9875
Dataset <i>VnAll</i>			
	Train	Val	Test
# of videos	9		1
# of frames	54865	4107	6373
Dataset <i>VnYT</i>			
	Train	Val	Test
# of videos	122		10
# of frames	303391	33063	20067

Table 1. Dataset statistics. Details of the three different subsets, the controlled setting data on a single string *VnI*, the controlled setting data on all strings *VnAll*, and the in-the-wild data on all strings *VnYT* and their respective train/val/test splits.

Finally, for all three subsets, we extract frames and pseudo ground-truth pitch using the spectral domain YIN algorithm [19] using the implementation in the *aubio* library (yinfilt) [20].

In addition to the above datasets, we consider the subset of the URMP dataset that contains videos of violin playing. We loosely crop the frames around the violinist and leave out 4 videos (single-instrument tracks) for testing and take the remaining violin videos for training and validation. This results in about 35000 frames for training and 12761 for testing. This dataset contains ground-truth pitch information. Therefore, we can train with actual ground-truth pitch. We consider a range of 33 semitones (MIDI numbers between 55 and 88). All models are trained and tested with the same train/val/test split on each dataset.

Furthermore, we generate STFT spectrograms for all mentioned datasets in order to train the audio to pitch network.

## 5. EXPERIMENTS

In this section, we evaluate both the audio to pitch (teacher), and the video frame to pitch (student) models. We consider using a single versus multiple input video frames. We first train the audio to pitch network to regress pitch from spectrograms. This network then serves as the teacher network when training the single frame to pitch network or the multi-frame to pitch network.

The models are trained in PyTorch [21] using the Adam optimiser [22] with an initial learning rate of 0.001. The learning rate is divided by a factor of 10 when the loss on the validation set plateaus. The batchsize is  $N = 64$  for the single frame architecture and the audio to pitch network, and  $N = 24$  for the architecture with 5 input frames. The frames are resized to  $400 \times 200$ . For the datasets *VnI* and *VnAll*, the frames are consistently more tightly cropped around the instrument whereas there is much more variation of the location and relative size of the instrument in *VnYT*.

**Evaluation measures.** We report the performance of our

Network	RPA	RPA tol	PA	ACA	ACE
Dataset <i>VnI</i>					
Audio to pitch	98.30	99.14	96.74	86.03	0.06
Frame to pitch	89.98	91.57	62.41	51.64	0.45
5 fr. to pitch (3D conv)	93.8	94.91	66.7	58.75	0.43
Dataset <i>VnAll</i>					
Audio to pitch	94.26	94.40	90.87	94.33	0.06
Frame to pitch	74.17	75.55	47.48	33.3	2.50
5 fr. to pitch (3D conv)	77.24	78.98	50.33	41.66	1.65
Dataset <i>VnYT</i>					
Audio to pitch	98.30	99.14	96.74	86.03	0.06
Frame to pitch	44.3	51.37	33.18	45.2	2.50
5 fr. to pitch (3D conv)	62.5	67.89	48.44	51.77	2.34
Dataset URMP					
Audio to pitch	98.28	98.5	96.73	98.88	0.07
Frame to pitch	53.11	58.3	42.71	39.86	2.73
5 fr. to pitch (3D conv)	57.3	62.04	45.26	41.79	2.43

Table 2. Evaluation of our models determining the accuracy in predicted pitch for the *VnI*, *VnAll*, *VnYT*, and URMP test sets. Higher is better for Raw Pitch Accuracy (RPA), Raw Pitch Accuracy with a tolerance of one frame (RPA tol), Pitch Accuracy (PA), and Average Class Accuracy (ACA). Lower is better for Average Class Error (ACE). Using multiple input frames improves the performance.

models in Table 2. For Raw Pitch Accuracy (*RPA*), a predicted pitch is counted as correctly estimated if it lies within one semitone of the ground truth pitch. *RPA tol* additionally allows the prediction to be off by at most one frame. Furthermore, we report Pitch Accuracy (*PA*) and Average Class Accuracy (*ACA*). *ACA* gives the averaged per-pitch-class accuracy. The *ACE* describes the average error between the predicted and the ground truth pitch class (*ACE* of 1 corresponds to an average error of one semitone).

**Video to pitch performance.** The audio to pitch teacher networks reach an *RPA* of above 90% on the test sets. This serves as a very good starting point to train the student frame to pitch networks. It can be observed that our method performs best when trained and tested on the simpler dataset with minimal ambiguities *VnI* and then *VnAll*. This corresponds with the intuition that this set-up is easier, as the fingers and therefore the pitch is more clearly visible at higher resolution as compared to *VnYT* or URMP. Nevertheless, a *RPA* of 62.5% for the frame to pitch network on the in-the-wild YouTube video dataset *VnYT* means that the pitch is estimated within a semitone of the ground-truth on average in 62.5% of the test cases; this verifies that our method generalises to unseen videos and people at test time on challenging ‘in-the-wild’ videos. When allowing for an offset of one frame in the predictions, we achieve an accuracy of 67.89% (*RPA tol*). This accounts for the case that the alignment between audio and visual information might not be perfect in the data which is the case for some of the downloaded videos. The reported lower performance on the URMP dataset may be due to the lower resolution size of the frames in the dataset and the limitations in terms of its dataset size which confirms the benefits of using our datasets to address this task. These results are impressive, given that our method estimates the pitch from visual information only and in unconstrained recording conditions. However, our method can only predict one pitch playing

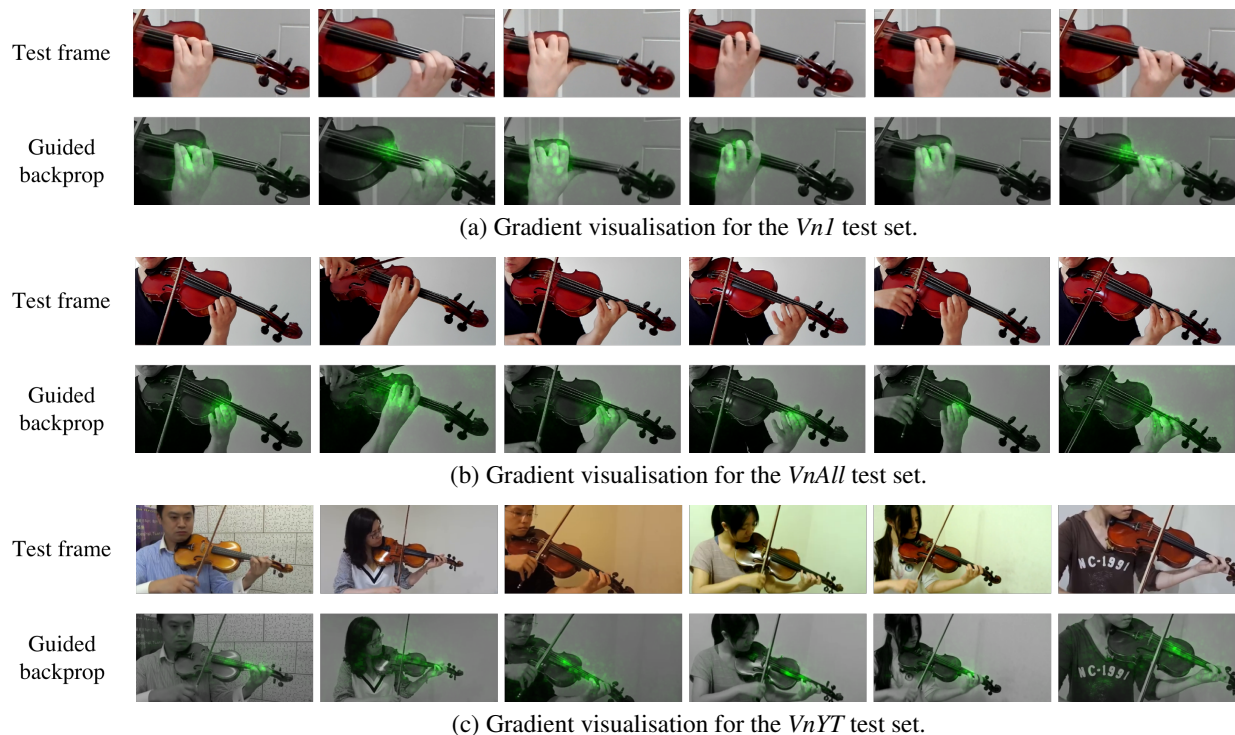


Figure 4. Discriminative information visualisations using guided backpropagation [23] for the test sets of the *VnI* subset in (a), the *VnAll* subset in (b), and the *VnYT* subset in (c). Heat maps are overlaid in the second rows of (a), (b), and (c). As demonstrated, the networks focus on the left hand across all the test frames even though the hands are in different positions relative to the frame. In (c), it can be seen that the network also seems to be focussing on the strings implying that it may be using vibrations or the movement of the strings in order to estimate pitch. The location of the instrument and strings relative to the left hand might serve as a further cue for estimating pitch.

at a time and cannot identify chords as it has been trained only on monophonic data.

Another interesting point is that there is a consistent improvement when using multiple frames as opposed to a single one as input to the frame to pitch network. This can be seen very clearly for the *VnYT* dataset (RPA tol of 67.89% vs. 51.37%). This is presumably due to the fact that visually it can be hard to determine just from the fingers of the left hand which string a note is played on. To solve this problem, the network needs to determine which string is active (using for example information from the bowing hand / bow or from the vibration of the strings). While the placement of the hand should be visible from a single image, the vibration of the string is unlikely to be visible (unless there is significant motion blur) without taking into account more frames.

**Visualizing what has been learnt.** To gain an insight into what the networks have learnt and how they infer the pitch from a given frame, we apply guided backpropagation [23] to our trained networks to determine which parts of the images are most discriminative. As demonstrated in Figure 4, the networks have learnt that the fingers of the left hand and the left hand itself are most relevant for predicting the pitch given a still frame. Potentially the network also makes use of some information about the vibration of the played strings (e.g. by recognising motion blur

around strings that are vibrating). This confirms that the networks do not simply memorise parts of a video, but instead learn to localise the left hands/fingers in the image in order to estimate pitch. However, the image regions which the networks focus on are actually quite small relative to the image size.

## 6. CONCLUSION

We have presented a method for addressing monophonic visual pitch estimation; given video frames of violin playing, our method can automatically estimate the pitch being played using *vision* alone. The presented task is extremely challenging, as it requires making use of subtle visual cues (such as the placement of the hand or string vibrations over the course of multiple frames), yet our network shows convincing results in three different scenarios: when only one string is played or all strings are played but the person and environment remains the same, and in unconstrained ‘in-the-wild’ videos. Moreover, our method is generalisable, as training the networks did not require any manual annotations; instead, the pseudo ground-truth pitch information was extracted automatically from the audio data. It will be interesting to use this framework to improve pitch prediction using both visual and audio information. This could prove useful when the audio is of poor quality. In addition to that, estimating pitch from vision might help the task

of sound source separation when similar instruments are played on. Furthermore, this method could be pushed further to estimating polyphonic violin music played on the same instrument.

### Acknowledgments

This work is supported by the EPSRC programme grant Seebibyte EP/M013774/1: Visual Search for the Era of Big Data. We are very grateful to Yael Moses for insightful discussions. We thank Arsha Nagrani for feedback.

### 7. REFERENCES

- [1] C. Bucilua, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *2th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.
- [2] J. Chung and A. Zisserman, "Out of time: automated lip sync in the wild," in *Workshop on Multi-view Lip-reading, ACCV*, 2016.
- [3] R. Arandjelović and A. Zisserman, "Objects that sound," in *Proc. ECCV*, 2018.
- [4] H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba, "The sound of pixels," in *Proc. ECCV*, 2018.
- [5] Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," in *NIPS*, 2016.
- [6] L. Chen, S. Srivastava, Z. Duan, and C. Xu, "Deep cross-modal audio-visual generation," in *Proceedings of the on Thematic Workshops of ACM Multimedia*, 2017.
- [7] A. Davis, M. Rubinstein, N. Wadhwa, G. Mysore, F. Durand, and W. T. Freeman, "The visual microphone: Passive recovery of sound from video," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 2014.
- [8] W.-L. Hao, Z. Zhang, and H. Guan, "Cmcgan: A uniform framework for cross-modal visual-audio mutual generation," *CoRR*, 2018.
- [9] A. Owens, P. Isola, J. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman, "Visually indicated sounds," in *Proc. CVPR*, 2016.
- [10] A. Ephrat, T. Halperin, and S. Peleg, "Improved speech reconstruction from silent video," in *ICCV workshop*, 2017.
- [11] T. Afouras, J. S. Chung, and A. Zisserman, "The conversation: Deep audio-visual speech enhancement," in *INTERSPEECH*, 2018.
- [12] A. Ephrat, I. Mosseri, O. Lang, T. Dekel, K. Wilson, A. Hassidim, W. T. Freeman, and M. Rubinstein, "Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation," *Proc. ACM SIGGRAPH*, 2018.
- [13] B. Li, C. Xu, and Z. Duan, "Audiovisual source association for string ensembles through multi-modal vibrato analysis," *Proc. Sound and Music Computing (SMC)*, 2017.
- [14] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications," *IEEE Transactions on Multimedia*, 2019.
- [15] E. Gómez Gutiérrez, P. Arias Martínez, P. Zinemanas, and G. Haro Ortega, "Visual music transcription of clarinet video recordings trained with audio-based labelled data," in *ICCV workshop*, 2017.
- [16] B. Zhang, J. Zhu, Y. Wang, and W. K. Leow, "Visual analysis of fingering for pedagogical violin transcription," in *Proceedings of the 15th ACM international conference on Multimedia*, 2007.
- [17] S. Goldstein and Y. Moses, "Guitar music transcription from silent video," in *Proc. BMVC.*, 2018.
- [18] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. BMVC.*, 2014.
- [19] P. Brossier, "Automatic annotation of musical audio for interactive systems," Ph.D. dissertation, Ph. D. thesis, Queen Mary University of London, London, UK, 2006.
- [20] P. Brossier, M. Hermant, E. Müller, N. Philippsen, T. Seaver, H. Fritz, and S. Alexander, "aubio/aubio: 0.4.6," Oct 2017. [Online]. Available: <https://doi.org/10.5281/zenodo.1002162>
- [21] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," 2017.
- [22] D. Kingma and J. Ba, "Adam: A method for stochastic optimization." *Proc. ICLR*, 2015.
- [23] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *ICLR workshop*, 2015.

# MININGSUITE: A COMPREHENSIVE MATLAB FRAMEWORK FOR SIGNAL, AUDIO AND MUSIC ANALYSIS, ARTICULATING AUDIO AND SYMBOLIC APPROACHES

Olivier Lartillot

RITMO Centre for Interdisciplinary Studies  
in Rhythm, Time and Motion  
University of Oslo  
olivier.lartillot@imv.uio.no

## ABSTRACT

The *MiningSuite* is a free open-source and comprehensive Matlab framework for the analysis of signals, audio recordings, music recordings, music scores, other signals such as motion capture data, etc., under a common modular framework. It adds a syntactic layer on top of Matlab, so that advanced operations can be specified using a simple and adaptive syntax. This makes the Matlab environment very easy to use for beginners, and in the same time allows power users to design complex workflows in a modular and concise way through a simple assemblage of operators featuring a large set of options. The *MiningSuite* is an extension of *MIRtoolbox*, a Matlab toolbox that has become a reference tool in MIR.

## 1. DESCRIPTION

The *MiningSuite*<sup>1</sup> is an open source Matlab toolbox composed of a large set of modules corresponding to the different possible types of signal processing representations and audio and music descriptors. These modules are structured into packages related to the different domains of study: signal processing (*SigMinr* package), auditory modelling (*AudMinr*), music analysis (*MusMinr*), video analysis (*VidMinr*), physics and motion analysis (*PhyMinr*), sequence processing (*SeqMinr*) and pattern mining (*PatMinr*).

Thanks to an innovative syntactic layer, both powerful and user-friendly, designed on top of Matlab, these modules can be easily applied to particular files or batch of files, and the numerous options available for each module can be modified. Modules can be connected and form data flow graphs. As such, complex design of set of audio or music analysis operations can be written in a very concise way through a simple assemblage of modules. They can be applied to large batches of files as well as to long files without memory issues thanks to implicit signal chunking and concatenation mechanisms. Another syntactic layer within

the operators' Matlab code enables to simplify and clarify the code. As the internal representation of signals integrates various types of decomposition (into frames, channels, segments) within a unified framework, the modules can adapt automatically to these various types of input.

Audio and symbolic representations and processes are tightly interconnected: The same type of symbolic representation is used to represent discrete constructions inferred from audio representation (such as peaks, segments, onset locations) as well as actual symbolic sequences (such as scores and MIDI sequences). Operators dedicated to high-level musical features extraction (key estimation, tempo, etc.) integrate signal processing, statistical and symbolic-based methods, and can be applied to both symbolic input and audio input (adding automated transcription steps wherever necessary).

The integration of expertise developed in separate areas of study into common modules encourages further reuse of these individual methods and their intermingling into a common framework.

*MiningSuite* is the official continuation of *MIRtoolbox* [1]. The architecture of the toolbox is much simpler, allowing faster computation and more transparent and clear code. Series of operations can be designed more efficiently and easily. Any signal can be imported and represented as an object of classes available in the *MiningSuite*. Each result also stores the complete description of the list of operations with all the specified options and parameters. Matrices imported into, used in, and exported from the *MiningSuite* have their internal structure clarified: the role of each dimension is made explicit using a systematic formalism.

## Acknowledgments

This work was partially supported by the Research Council of Norway through its Centres of Excellence scheme, project number 262762. This work was also partially supported by a research fellowship granted by the Academy of Finland during the years 2009–2014.

## 2. REFERENCES

- [1] O. Lartillot and P. Toiviainen, "Mir in matlab (ii): A toolbox for musical feature extraction from audio," in *International Conference on Music Information Retrieval*, 2007.

<sup>1</sup> <http://olivierlar.github.io/miningsuite/>



# DRAWING GEOMETRIC FIGURES WITH BRAILLE DESCRIPTION THROUGH A SPEECH RECOGNITION SYSTEM

Africa Chamorro, Ana M. Barbancho, Isabel Barbancho, Lorenzo J. Tardón

Universidad de Málaga, Andalucía Tech, ATIC group, E.T.S.I. Telecomunicación,

Dpt. Ingeniería de Comunicaciones, Campus Teatinos s/n, 29071 Málaga

abp@ic.uma.es, ibp@ic.uma.es, lorenzo@ic.uma.es

## ABSTRACT

In this contribution, a system that represents drawings of geometric figures along with their description transcribed in Braille controlled by means of commands acquired by a speech recognition scheme is presented. The designed system recognizes the spoken descriptions needed to draw simple geometric objects: shape, colour, size and position of the figures in the drawing. The speech recognition method selected is based on a distance measure defined with Mel Frequency Cepstral Coefficients (MFCCs). The complete system can be used by both people with visual and with hearing impairments thanks to its interface which, in addition to showing the drawing and the corresponding transcription in Braille, also allows the user to hear the description of commands and final drawing.

## 1. INTRODUCTION

Nowadays, there are many voice recognition systems, such as the successful Siri. However, they still may be unsatisfactory for people with certain impairments because the use of just voice and sound may be insufficient. In the world there are millions of people who suffer some kind of communicative disability, and defining a common language for all of them is a really complex task. In this context, Braille is a valid and effective tool for both people with visual and hearing impairments, as it helps them to receive and understand the information of the world around them [1]. In this contribution, a system that creates drawing with geometric figures along with their transcription in Braille is presented. The drawings are built by means of commands issued by means of a speech recognition approach. The specificity of this system is the inclusion of both speech recognition and Braille transcription schemes.

## 2. SYSTEM DESCRIPTION

The general structure of the system developed is shown in Fig. 1. In this figure, the two different parts that compose the system can be observed, specifically: the speech recognition subsystem and the drawing with Braille transcription

scheme. The whole system has been implemented in Matlab.

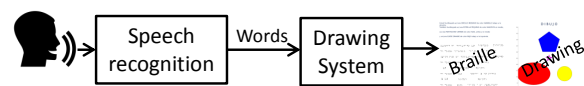


Figure 1. General structure of the developed system.

The two parts of the system developed are described next.

### 2.1 Speech recognition subsystem

Among the different speech recognition schemes that can be employed, a simple approach based on *MFCCs* has been selected for the tool developed. *MFCCs* are used in different audio analysis applications, including speech recognition [2]. Fig. 2 shows a schematic of the speech recognition subsystem. In this figure, it can be observed that this subsystem has three different parts: pre-processing, *MFCC* estimation and recognition.

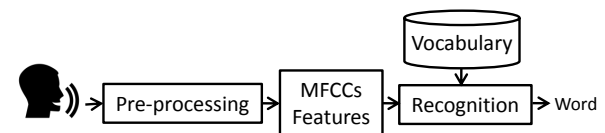


Figure 2. Speech recognition subsystem.

The pre-processing stage is aimed at preparing the voice to perform the recognition. This block includes three stages: noise reduction, pre-emphasis and segmentation with windowing. Hamming windows have been employed because of the fast decay of their side lobes [3]. Then, *MFCCs* are calculated using the available Matlab function. 40 Mel filter are used when after the *DCT*, the first 36 coefficients are selected (the first one is not employed). Spoken command recognition is implemented by simply measuring the geometric distance between the *MFCCs* extracted and the pre-calculated *MFCCs* of the vocabulary used by the system (see Table 1). The *MFCCs* of the vocabulary are calculated as the arithmetic mean of 10 different recordings of each word.

### 2.2 Drawing with Braille transcription subsystem

The general structure of the Drawing with Braille transcription subsystem is presented in Fig. 3. In each screen

Copyright: © 2019 Africa Chamorro, Ana M. Barbancho, Isabel Barbancho, Lorenzo J. Tardón et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Geometry	Colour	Size	Position
Círculo	Amarillo	Pequeño	Arriba derecha
Cuadrado	Magenta	Mediano	Arriba centro
Rectángulo	Cian	Grande	Arriba izquierda
Elipse	Rojo		Centro derecha
Triángulo	Verde		Centro centro
Pentágono	Azul		Centro izquierda
Estrella	Negro		Abajo derecha
	Blanco		Abajo centro
			Abajo izquierda

Table 1. Vocabulary used by the system (in Spanish).

of the developed application, the users have at their disposal visual and hearing resources as well as the possibility of keyboard interaction and the repetition of the speech command. As shown in Fig. 3, in order to draw a geometric figure, position, size, geometry and colour must be said sequentially. Users can draw as many figures as they want. Fig. 4, shows the results of a drawing with its description in Spanish and Braille.

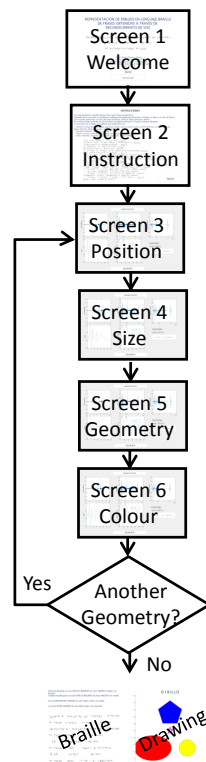


Figure 3. Flow diagram of the developed system.

### 3. SYSTEM EVALUATION

Performance tests have been carried out with 10 different users, who recorded all the vocabulary words in two different scenarios: with and without the possibility of command repetition. The number of repetitions was limited to three. Table 2, shows the results of the speech recognition system implemented with and without repetition. These results in-

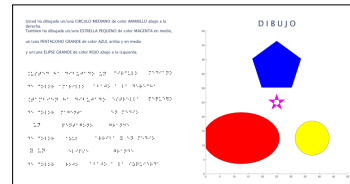


Figure 4. Illustration of a drawing done with the system developed.

dicates that repeating the same word greatly improves the recognition system, as expected. Note that although the accuracy in the detection of words can be improved, the users said they were pleased with its operation.

Vocabulary	One repetition	Three repetitions
Position	51.10%	74.45%
Size	70.00%	86.64%
Geometry	60.00%	80.00%
Colour	61.25%	83.75%

Table 2. Word recognition evaluation results.

### 4. CONCLUSIONS

A system that draws geometrical figures, along with their transcription in Braille on the basis of commands given through a speech recognition scheme has been presented. The designed system recognizes simple spoken descriptions needed to draw geometric objects and their simplified location. The evaluation of speech recognition method developed shows that repeating the words to create the models for the later detection greatly improves the recognition performance, which was sufficient to make the subjects pleased with its operation. The system can be used by people with visual impairments and with hearing impairments as the interface includes Braille transcription and hearing aids.

### Acknowledgments

This work has been funded by the Ministerio de Economía y Competitividad of the Spanish Government under Project No. TIN2016-75866-C3-2-R. The work has been done at Universidad de Málaga, Campus de Excelencia Internacional Andalucía Tech.

### 5. REFERENCES

- [1] T. Choudhary, S. Kulkarni, and P. Reddy, "A braille-based mobile communication and translation glove for deaf-blind people," in *International Conference on Pervasive Computing (ICPC2015)*, 2015, pp. 1–4.
- [2] L. Tardón, S. Sammartino, and I. Barbancho, "Design of an efficient music-speech discriminator," in *Acoustic Society of America*, 2010, pp. 271 – 279.
- [3] A. Oppenheim and R. Schaffer, *Discrete-Time Signal Processing*. Pearson New International Edition, 2014.

# INTERACTIVE MUSIC TRAINING SYSTEM

**Daniel Moreno, Isabel Barbancho, Ana M. Barbancho, Lorenzo J. Tardón**

Universidad de Málaga, Andalucía Tech, ATIC Research Group, E.T.S.I. Telecomunicación,

Dpt. Ingeniería de Comunicaciones, Campus Teatinos s/n, 29071 Málaga

ibp@ic.uma.es, abp@ic.uma.es, lorenzo@ic.uma.es

## ABSTRACT

In this contribution, we present an interactive system for playing while learning music. The game is based on different computer games controlled by the user with a remote control. The remote control has been implemented using inertial measurement sensors (IMU) for 3D tracking. The computer games are programming in Python and allow to practice rhythm as well as the tune, ascending or descending, of musical notes.

## 1. INTRODUCTION

The serious game concept is used to describe games designed to serve an additional purpose to that of pure entertainment. The term serious game had been introduced in 1970 [1] but it was not until early 2000s when they surge in different types of educational games designed for the younger learner.

Serious games for music learning are very interesting [2], especially for children who start to learn music, given the difficulty involved in the individual study of music. Among the different elements to practice in the music studio, rhythm and tone perception are basic and general to any kind of music. In this contribution, we present an interactive music training system that allows to practice rhythm and tone perception in a fun and easy way, using a remote control and a computer game.

## 2. SYSTEM DESCRIPTION

The scheme of the interactive music training system is shown in Fig. 1. In this figure, it can be seen that the developed system consists of two different parts: a remote control and a computer game module. The remote control is designed using inertial measurement sensors (IMU) and the computer game system is programmed in Python.

### 2.1 Remote control subsystem

The functionality of the remote control includes to communicate with the computer and detect the position and movement of the user's hand. Fig. 2 shows the block diagram of the remote control hardware.

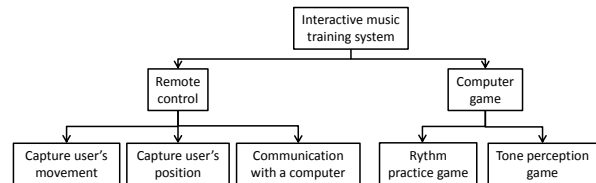


Figure 1. Diagram of the interactive music training system.

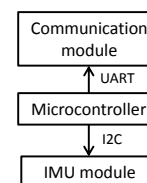


Figure 2. Block diagram of the remote control hardware.

In Fig. 2, the tree basic modules of the remote can be observed:

- **Microcontroller:** This module is the core of the remote control system; its function is to calculate and send the information of hand position and movement of the users to a computer. Texas Instrument TM4C123GH6PM is the microcontroller selected because it has enough memory to include a real time operating system for concurrent processing.
- **Communication module:** Bluetooth HC-06 has been selected to send information to the computer by means of Bluetooth 2.0.
- **IMU module.** This module is used to get measures of acceleration in the three axes. These measures are processed in the microcontroller to obtain the position and movement of the user's hand. The intelligent sensor BNO055 has been selected because it is an inertial absolute orientation 9-axis sensor.

In Fig. 3 the remote control designed is presented. In this figure, it can be seen that it is a very compact system.



Figure 3. Remote control designed.

Copyright: © 2019 Daniel Moreno, Isabel Barbancho, Ana M. Barbancho, Lorenzo J. Tardón et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

## 2.2 Computer game subsystem

The computer game system is programmed in Python. In order to achieve the requirement of serving to practice rhythm as well as the tune, ascending or descending, of musical notes, three different games have been developed: 'Note order', 'Rhythm fun' and 'Virtual Drums'. Also, an utility to compose your own scores for the rhythm game has been developed.

### 2.2.1 Note order game

In this game, the user listens two notes sequentially with a separation of one second. Once the notes have been played, the user must choose whether the sequence of notes have ascending, descending or equal tone. The selection is made by moving the remote control, up, down or horizontally. Fig. 4 shows several screenshots of the note order game.



Figure 4. Note order game screenshots.

### 2.2.2 Rhythm fun game

In this game, the user has to set the rhythm of a score. The game includes several predefined scores but the user can create his own score. The user has two aids: an arrow that moves through the notes of the score and the sound of a metronome. Fig. 5 shows a screenshot of the rhythm fun game.

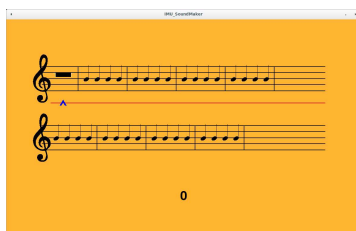


Figure 5. Rhythm fun game screenshot.

### 2.2.3 Virtual drum game

In this game, the user has the possibility of using the remote control as a drumstick of a drummer. Fig. 6 shows a screenshot of the virtual battery game.

### 2.2.4 Compose your own score

With this utility the user has the possibility of using the remote control and the keyboard to compose his own score to practice rhythm. In this case, the first screen allows to select the measure and the second one to compose the score. Fig. 7 show two screenshots of this tool.

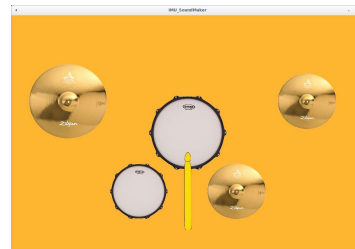


Figure 6. Virtual drum game screenshot.



Figure 7. Compose your own score screenshots.

## 3. CONCLUSIONS

An interactive system for playing while learning music has been presented. The game is based on different simple computer games controlled by the user with a specifically designed remote control. The remote control has been implemented using IMU sensors for 3D tracking. The computer games are programmed in Python and allow to practice rhythm as well as the tune progression, ascending or descending, of musical notes. The system has been used with different music students that were pleased with its operation.

### Acknowledgments

This work has been funded by the Ministerio de Economía y Competitividad of the Spanish Government under Project No. TIN2016-75866-C3-2-R. The work has been done at Universidad de Málaga, Campus de Excelencia Internacional Andalucía Tech.

## 4. REFERENCES

- [1] C. C. Abt, *Serious Games*. The Viking Press, 1970.
- [2] I. Barbancho, L. Tardón, and A. Barbancho, *Real-Time audio interaction in serious games for music learning*. 19th International Society for Music Information Retrieval Conference, 2018.



# COPYING CLAVE – A TURING TEST

Simon Blackmore

Sonic Art Research Unit – Oxford Brookes University  
[simon@simonblackmore.net](mailto:simon@simonblackmore.net)

## ABSTRACT

A blindfolded instructor (evaluator) plays a clave pattern. A computer captures and repeats the pattern. After 1 minute the experiment stops. This process is repeated by a human who also tries to copy the clave. After another minute they stop and the evaluator assesses both performances.

The demonstration will be presented in a lively informal way allowing people to adopt the role of either the player or instructor. It is hoped that it stimulates a debate of how the methods and technology could be further developed,

## DEMONSTRATION

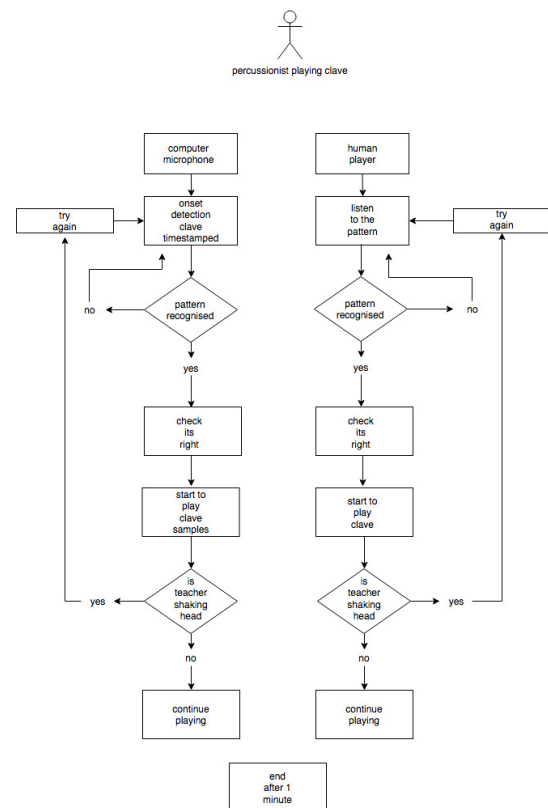
A clave is a rhythmic pattern used in many forms of music to provide a framework for musicians to play with. It is very common in Afro-Cuban music and typically played with two pieces of wood or metal to create a clear loud sound enabling it to be identified through the rest of the music.

For the purposes of this demonstration it can be any pattern that can be accurately repeated. For now we are not concerned with deriving a tempo from the pattern or where it starts and stops.

The instructor is blindfolded. A coin is tossed. Heads the machine plays first, tails the human plays first. The instructor begins to play clave.

On the machine's turn, the clave is captured through a microphone, onset times are recorded and an autocorrelation algorithm is used to identify the pattern. When a repeated pattern is identified, the clave pattern is silently checked against the incoming pattern from the instructor and then, if correct, played back to the instructor via samples through a speaker. A camera input and CV software allow for visual feedback from the instructor as to whether it is playing the pattern right or not. A shake of the head indicates that the pattern is played incorrectly and must stop.

On the human's turn, ears are used to listen to the clave and when the pattern is identified it is played back by lightly tapping the laptop touch pad, this again produces the sampled clave sound through the speakers. As before a shake of the head indicates that the pattern is played incorrectly and the player must stop and try again.



Copyright: © 2019 First author et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

# Resonance Improviser: A system for transmitting the embodied sensations of vocalization between two people during improvisation

Tejaswinee Kelkar  
University of Oslo  
author1@smcnetwork.org

Lynda Joy Gerry  
Aalborg University Copenhagen  
lyn@create.aau.dk

## ABSTRACT

This is a system prototype for joint vocal improvisation between two people that involves sharing embodied sensations of vocal production. This is accomplished by using actuators that excite two participants' rib cages with each other's voices, turning a person's body into a loud speaker. A microphone transmits vocal signals and the players are given a Max Patch to modulate the sound and feel of their voice. The receiver hears the other person's speech and effects through their own body (as if it were their own voice), while also feeling the resonance of the sound signal as it would resonate in the chest cavity of the other. The two players try to re-enact and improvise a script prompt provided to them while not knowing what the other person can hear, of their voice. The game may or may not turn collaborative, adversarial, or artistic depending on the game play.

## 1. INTRODUCTION

In this paper, we present the proof of concept for the Improvised Resonator, sketching its architecture, design, user experience, and the experimental methodology that will be used to evaluate the effects of the experience for the context of this demo.

### 1.1 Background Work

The Resonance Improviser is an example of an Augmented Social Embodiment (ASE) system. ASE combines features of sensory augmentation devices [1] with social embodiment [2] to use new sensor and wearable technologies to transmit and share aspects of our embodiment that cannot normally be shared. For example, by using haptic communication devices we can remotely transmit information about one person's muscle movements while performing a task to the associated muscles and joints of another person [3]. Using sensor technologies to enrich social interactions is related to projects such as "enriched social interactions" [4], "mediated intimacy" [5], "co-embodied technology" [6], "interpersonal biofeedback" [7], and "phatic technologies" [8]. Specifically, these related projects embed wearable technologies in social interactions with the

goal to enhance those interactions. However, very few of these projects use sensory augmentation wearable devices to transmit information about another person's embodied state. Many of these previous related projects represent information about another person in a very symbolic, abstract way. Instead, the Resonance Improviser directly transmits vibrations of the vocal cavity from one person to another.

### 1.2 User Experience

The users will be instructed to go to opposite corners of the room, facing away from one another, and looking up to follow a script that will be presented on a screen. Each player will also receive a screen to control a Max Patch to modulate their voice, with noise and glitch effects that the player themselves will not be able to experience before sending them to their partner. We adapted a script of human-computer interactions from Stanford University's colorful personality chatbot transcripts, with one player taking the human role and the other taking the computer role.

### 1.3 Hardware Setup

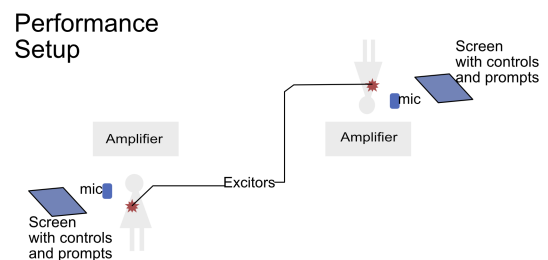


Figure 1. Schematic diagram of demo setup. The two participants, separated from each other take part in vocal improvisation, while wearing excitators that communicate their

Each participant can influence the way their voice will be heard and felt by their partner by adding various audio effects. They will transmit their voice with these audio effects to their partner's rib cavity without hearing their voice modulated by the effects for themselves.

The idea of this project is to create a disembodied voice installation through actuating two participants' rib cages with each other's voices. Two speakers in this interface

Copyright: © 2019 Tejaswinee Kelkar et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

act as resonating bodies producing each others voices. The two players are encouraged to play with loopers, vocoders, sampling and glitch to speak with/through each other, while they don't receive feedback about how their own production sounds. They also hear the other persons voice through their own rib cage cavity. This project allows users to transmit from one person to another an aspect of the inner embodied sensations created by vocal vibrations in the chest cavity. Using an actuator device, users can feel vibrotactile sensations on their own sternum and ribcage recorded from another's body while listening to this other person speak (and/or sing?).

## 2. CONCLUSIONS

This creates a novel experiential context wherein which something which is normally an intimate part of our social embodiment, our voice, is suddenly shared. Exciting your own rib cavity using someone else's voice is a body transfer experience. We are familiar with the corporeality and feel of our own voice and its inevitable resonance while breathing, speaking, singing, etc., and this makes experiencing another person's voice as vibration in your body an intimate experience. In essence, they hear the other person's voice in a way that they only ever hear their own voice and we are making interpersonal something which is normally intrapersonal. Embedding physiological sensors and biofeedback in social interactions has been shown to enhance interpersonal connections and intimacy among strangers (Bala et al., 2004; Vetere et al., 2005, Gibbs et al., 2005). In a 2014 review of strategies that designers can use to create technologies to foster interpersonal connection, Hassenzahl and colleagues suggest joint action as an effective strategy. Thus, we implemented joint vocal improvisation in our design.

## 3. REFERENCES

- [1] K. Kaspar, S. König, J. Schwandt, and P. König, "The experience of new sensorimotor contingencies by sensory augmentation," *Consciousness and cognition*, vol. 28, pp. 47–63, 2014.
- [2] L. W. Barsalou, P. M. Niedenthal, A. K. Barbey, and J. A. Ruppert, "Social embodiment," *Psychology of learning and motivation*, vol. 43, pp. 43–92, 2003.
- [3] A. Chellali, C. Dumas, and I. Milleville-Pennel, "Wyfiwif: A haptic communication paradigm for collaborative motor skills learning," in *Web Virtual Reality and Three-Dimensional Worlds 2010*. IADIS, 2010, pp. 301–308.
- [4] S. Bala, T. McDaniel, and S. Panchanathan, "Visual-to-tactile mapping of facial movements for enriched social interactions," in *2014 IEEE International Symposium on Haptic, Audio and Visual Environments and Games (HAVE) Proceedings*. IEEE, 2014, pp. 82–87.
- [5] F. Vetere, M. R. Gibbs, J. Kjeldskov, S. Howard, F. Mueller, S. Pedell, K. Mecoles, and M. Bunyan, "Mediating intimacy: designing technologies to support strong-tie relationships," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2005, pp. 471–480.
- [6] J. Van Dijk and R. Mitchell, "Co-embodied technology: a design space for human being," *TEI14 Work-in-Progress (poster)*, 2014.
- [7] J. M. Tennant, S. Cook, M. C. Moldoveanu, J. B. Peterson, and W. A. Cunningham, "Interpersonal resonance: Developing interpersonal biofeedback for the promotion of empathy and social entrainment," in *International Conference on Applied Human Factors and Ergonomics*. Springer, 2018, pp. 208–214.
- [8] M. R. Gibbs, F. Vetere, M. Bunyan, and S. Howard, "Synchromate: a phatic technology for mediating intimacy," in *Proceedings of the 2005 conference on Designing for User eXperience*. AIGA: American Institute of Graphic Arts, 2005, p. 37.





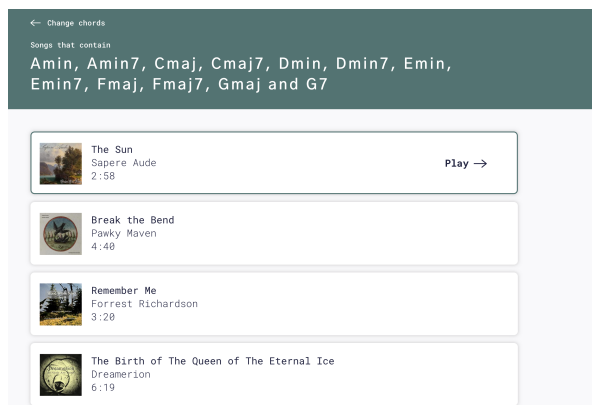


Figure 2. The list of results for a particular chord query.

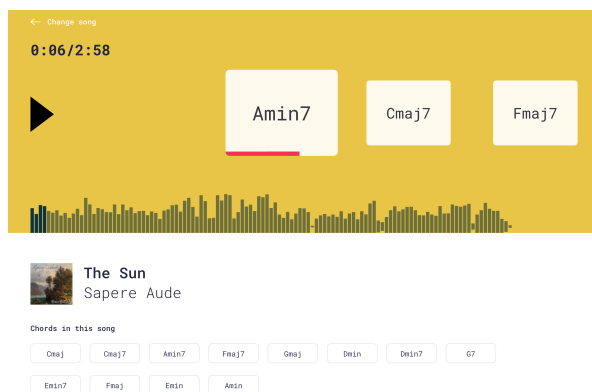


Figure 3. The music player displaying the chords in sync with the music.

## Acknowledgments

This work has been partly funded by the UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/L019981/1 and by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 688382.

## 3. REFERENCES

- [1] O. Celma, *Music Recommendation and Discovery: The Long Tail, Long Tail, and Long Play in the Digital Music Space*. Springer-Verlag Berlin Heidelberg, 2010.
- [2] W. B. de Haas, J. P. Magalhães, D. ten Heggeler, G. Bekenkamp, and T. Ruizendaal, "Chordify: Chord Transcription for the Masses," in *Proceedings of the 13th ISMIR Conference, Late Breaking and Demo Session*, 2014.
- [3] J. Pauwels, K. O'Hanlon, G. Fazekas, and M. B. Sandler, "Confidence Measures and Their Applications in Music Labelling Systems Based on Hidden Markov Models," in *Proceedings of the 18th ISMIR Conference*, 2017, pp. 279–285.
- [4] J. Pauwels, A. Xamb, G. Roma, M. Barthet, and G. Fazekas, "Exploring real-time visualisations to support chord learning with a large music collection," in *Proceedings of the 4th Web Audio Conference*, 2018.

# INTERNAL COMPLEXITY FOR EXPLORATORY INTERACTION

Mads Hoby

Roskilde University, Computer Science  
mads@hoby.dk

## ABSTRACT

When designing interactive sound for non-utilitarian ludic interaction, internal complexity can be a way of opening up a space for curiosity and exploration. Internal complexity should be understood as non-linear mappings between the input and the parameters they affect in the output (sound). This paper presents three different experiments which explore ways to create internal complexity with simple interfaces for curious exploration.

## 1. INTRODUCTION

This paper presents an exploration of the relations between physical and computational forms [1], [2]; how this affects the overall expression when designing for *exploration*. For us to do this we need to move past more utilitarian perspectives like affordance [3], transparency and efficiency, and instead consider factors more related to *ludic play* [4] like *curiosity* [5] and *ambiguity* [6].

Our basis for exploration lies in the searching for the *sweet spot* [7], [8] between chaos and predictability. We want people to be drawn by their own curiosity of not being able to decode the interaction pattern (chaos), while at the same time having a sense that their actions are the main contributor to the sounds (predictability). Specifically, we wonder if it is possible to forward curiosity and exploration by designing simple interfaces with relatively large non-trivial internal soundscapes.



**Figure 1:** Can a simple interface (tip of the iceberg) with a relatively complex internal logic (bottom of the iceberg) create a space for exploration and curiosity?

## 2. THE MACHINES

We designed three different standalone Arduino [9] based interactive noise machines. They all used simple inputs such as potentiometers and touch sensors. The Touchbox and the Complexicator used a home-made four voice wave-table synth and the Noise machine used a set of bit-shifting algorithms to produce the soundscapes. The mapping of input to sound generator differed greatly.

### 2.1 The touchbox

The Touchbox offers a play session for two participants at a time. The role of the technology is to sense physical bare skin connection between the participants. The sensing yields analogue values in a range starting from a few centimetres from actual touch, via light touch to full contact. The values are converted into a relatively complex soundscape, which is played back to each participant through their headphones.

Based on the analogue touch value, activity (change) and contact over time (incremental value) is derived. The three parameters are mapped to different dimensions in the soundscape (pitch and modulation on four voices). This creates a sense of a multi-dimensional interface for a body to body interaction. E.g. kissing, stroking, tapping, grabbing etc. give different sonic results.



**Figure 2:** The touchbox consists of a wooden box with a meter and a light bulb.

### 2.2 Algorithmic Noise Machine

Bit-shifting can be used as an alternative way of creating "music". The principle yields rather unpredictable results where small changes in the bit-shifting algorithm can have a large consequence for the produced soundscape.

The different parameters are changed through the four potentiometers. One potentiometer controls the current

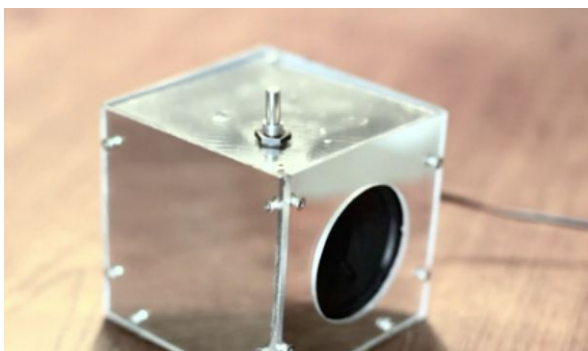
algorithm and the other potentiometers change the parameters for the algorithm. Although the sounds can be considered crude it is quite engaging to experiment and play with, even for the designer, who cannot predict the possibilities himself.



**Figure 3:** The noise machine has four potentiometers, without any instructions of how these modulate the sounds.

### 2.3 The Complexicator

The Complexicator is an experiment in mapping as many parameters as possible from a four-voice wavetable synth onto one potentiometer. A fast turn switches the potentiometer's role to control different sonic parameters (e.g. voice, pitch, pattern and type of wavetable). A slow turn then changes the value of the specific parameter. Technically speaking there would be a concrete coupling at all times, but in practice, the interface was overly complex and hard to grasp. This gave a sense of powerlessness while still having a sense of having some "say" in the output.



**Figure 4:** The Complexicator has just one potentiometer to change the internal four voices and their parameters.

## 3. CONCLUSION

The three different experiments are thought of as exemplary artefacts [10], [11]. Their intention is to extend space for exploration and curiosity. The different prototypes have been tested in various degrees and our preliminary findings point towards the following: The intimacy of touching other people with the *Touchbox* resonates well in such a way that the internal logic becomes an excuse to interact. The noise machine creates a sense of control and exploration. One can return to a previous setting by dialling the knobs back to a previous setting. The uncontrollable element of turning speed in the Complexicator gives a sense of random exploration; as if the box has a personality of its own.

## Acknowledgments and sources

Touch sensing by Nikolaj Møbius:

<http://fablab.ruc.dk/sensing-touch-with-arduino/>

8 Bit wavetable synth for Arduino by Nikolaj Møbius:

<https://www.instructables.com/id/Turn-your-Arduino-into-a-4-voice-wavetable-synth-w/>

Bit Shifting for sound generation by multiple experimenters: Duane Banks, Viznut & Tejeez:

<https://www.instructables.com/id/Algorithmic-noise-machine/>

<http://rcarduino.blogspot.com/2012/09/algorithmic-music-on-arduino.html>

## 4. REFERENCES

- [1] L. Hallnäs, "On the foundations of interaction design aesthetics: Revisiting the notions of form and expression," *International Journal of Design*, vol. 5, no. 1, pp. 73–84, 2011.
- [2] A. Vallgård, "Giving form to computational things: developing a practice of interaction design," *Pers. Ubiquit. Comput.*, vol. 18, no. 3, pp. 577–592, Mar. 2014.
- [3] D. A. Norman, "Affordance, conventions, and design," *Interactions*, vol. 6, no. 3, pp. 38–43, May 1999.
- [4] B. Gaver, "DESIGNING FOR HOMO LUDENS, STILL," pp. 1–17, Apr. 2008.
- [5] R. Tieben, T. Bekker, and B. Schouten, "Curiosity and interaction: making people curious through interactive systems," in *Proceedings of the 25th BCS Conference on Human-Computer Interaction*, 2011, pp. 361–370.
- [6] W. W. Gaver, J. Beaver, and S. Benford, "Ambiguity as a resource for design," in *Proceedings of the conference on Human factors in computing systems - CHI '03*, Ft. Lauderdale, Florida, USA, 2003, p. 233.
- [7] W. Gaver, J. Bowers, T. Kerridge, A. Boucher, and N. Jarvis, "Anatomy of a failure: how we knew when our design went wrong, and what we learned from it," in *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, Boston, MA, USA, 2009, p. 2213.
- [8] M. Hoby, *Designing for Homo Explorens: open social play in performative frames*. Faculty of Culture and Society Malmö University, 2014.
- [9] Arduino, "Home page." 999999.
- [10] B. Gaver and J. Bowers, "Annotated portfolios," *Interactions*, vol. 19, no. 4, p. 40, Jul. 2012.
- [11] J. Bowers, "The Logic of Annotated Portfolios: Communicating the Value of 'Research Through Design.'"

# ADAPTIVE BODY MOVEMENT SONIFICATION IN MUSIC AND THERAPY

**Christian Bauman**

Hochschule Fulda

christian.baumann@pg.hs-fulda.de

**Johanna Friederike Baarlink**

Musikschule Fulda

johanna.baarlink@gmail.com

**Jan-Torsten Milde**

Hochschule Fulda

milde@hs-fulda.de

## ABSTRACT

In this paper we describe the ongoing research on the development of a body movement sonification system. High precision, high resolution wireless sensors are used to track the body movement and record muscle excitation. We are currently using 6 sensors. In the final version of the system full body tracking can be achieved. The recording system provides a web server including a simple REST API, which streams the recorded data in JSON format. An intermediate proxy server pre-processes the data and transmits it to the final sonification system. The sonification system is implemented using the web audio api. We are experimenting with a set of different sonification strategies and algorithms. Currently we are testing the system as part of an interactive, guided therapy, establishing additional acoustic feedback channels for the patient. In a second stage of the research we are going to use the system in a more musical and artistic way. More specifically we plan to use the system in cooperation with a violist, where the acoustic feedback channel will be integrated into the performance.

## 1. INTRODUCTION

Real time measurement of human body movement provides an excellent technical basis for a larger number of application and research scenarios. In this paper we describe the ongoing design and development of a real time sonification system for body movement data. Two application contexts for this system have been defined by us:

1. body movement sonification as an additional bio feedback channel as part of a physio therapy in multiple settings
2. body movement sonification as an additional channel as part of a musical performance, in our case playing the viola by our second co author, a trained musician and music teacher

So far, we focused on the first context, as we are still in the technical setup phase of our research.

Guided movements, which are body to body interactions between patient and therapist, are a central, important means

for the treatment of a large number of human illnesses. The therapist is guiding the patient with her body movements and helps him, by giving language instructions, is controlling the tempo of the movement, controls the intensity of the patients movement and defines the rhythm of the movement. By holding the hand or touching the arm, a helpful supporting haptic feedback is given to the patient, who in turn, is reacting to the interaction and thus adapts and improves his body movements leading to a better recovery.

The similarity between therapy and musical performance or dance should be rather obvious now. In both cases we find the same pre conditions: dance consists of a body to body interaction, musical performance consists of a body to instrument interaction.

The integration of body movement and sound production is an integral part of the musical and artistic expression. Musicians use body movements in multiple ways, obviously to create the sound in conjunction with their instruments, but also to intensify the musical effect, e.g. by synchronizing their body movements with the rhythm or the dynamic of the performed musical piece.

Body movements are also used to communicate during musical performance. Spoken language is often not to be used during a musical performance, giving posture, gesture, mimic and gaze a more important and prominent communicative function.

### 1.1 Bio feedback in therapy

Bio feedback training (BFB) is a powerful means to learn and re-learn body motion patterns. It is often used with patients suffering from neuro muscular disorders or pain symptoms of the motion apparatus. A positive effect through pallesthetic BFB can also be recognized for children with innate cerebral pases ([1]). BFB is also used for patients with strokes and facial pases ([3]; [8]; [2]). Current BFB systems are focusing on visual bio feedback (see [6]). These systems require the patient to visually focus an external source, quite often a monitor displaying some kind of visual stimulus. This is interfering with the patients ability to perceive their own body movements, which could be a very important source of information during therapy. It is currently not clear, whether audio based BFB systems provide better results for the therapy of motion limited or disabled patients, ([7]; [4]) And it is also unclear, how a satisfactory sonification process for body movement data could be implemented within the context of physio therapy ([5]).

Copyright: © 2019 First author et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



## 2. THE MEASURING SYSTEM

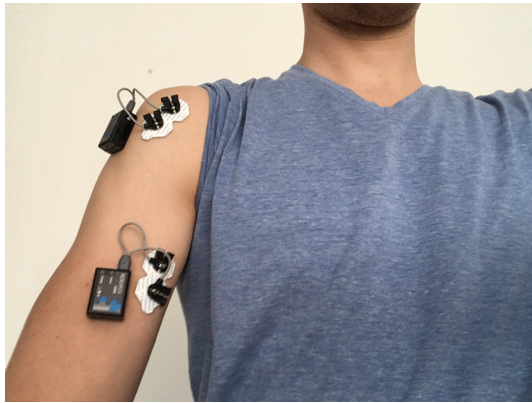


Figure 1. The experimental setup: sensors can be placed anywhere on the human body. In our case, we are interested in arm movements, either in a therapeutic context, in a later stage, measuring the arm movements of a violinist.

Within our system, the recording of the body movement is conducted using wireless sensors, that will be attached to specific body parts of the musician/patient. These sensors provide high quality readings with a high temporal resolution. The maximum sample rate of the sensors is at 3000 Hz, giving detailed information on the muscle excitation, the acceleration and the spatial alignment. The used Noraxon software provides the functionality to synchronize audio- and video streams with the recorded body movement data. In addition, a number of simple statistical steps can be computed by the software. It also visualizes the measurements in a live graph and is able to create a very simple auditory feedback, based on the definition of threshold values for the parameters.

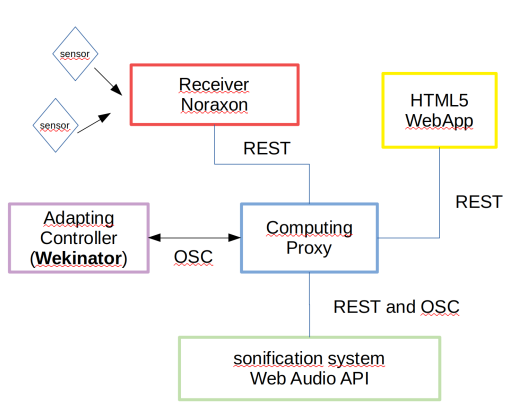


Figure 2. The system architecture: data is transmitted using a REST API. The computing proxy is preprocessing the incoming data and uses the wekinator to adapt the sonification parameters. The sonification process is implemented as a web audio based system running in the client browser (and also in SuperCollider).

In order to access the live data of a recording, the Noraxon software provides the user with a built-in web server

that is implementing a simple REST API. The measurements are streamed as raw data and are encoded as a JSON compatible string. The blocksize of these data chunks is variable and depends on the request interval of the connected client system, in our case, the computing proxy system.

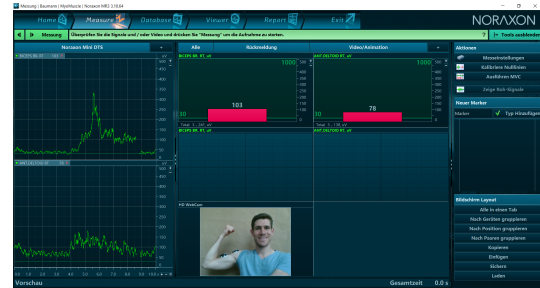


Figure 3. The Noraxon software is recording, audio data, video data and the body movement data in real time. Data is displayed as a live oscillogram.

The computing proxy system parses the data blocks and performs a set of pre-processing steps. It is implemented in python and uses powerful libraries to extract a number of key figures from the raw data. These include gliding average and standard deviation. It also integrates the signal and performs other standard statistical computations. Through these steps, the computing proxy reduces the amount of data, that is sent to the sonification system.

## 3. SONIFICATION WITH THE WEB AUDIO API

The sonification system has been implemented as a web based application using the web audio API for creating the sound<sup>1</sup>. The central component of the system is a 3 oscillator subtractive synthesizer. Its internal architecture is fixed (non modular) and follows the standard approach taken by most of the current analogue and digital synthesizers, consisting of a chain of VCO, VCF and VCA. In addition LFOs and an EG are provided to modulate a number of parameters of the main components (e.g. frequency, pulse width, filter cutoff frequency and filter resonance). In order to receive the movement data from the computing proxy, a simple timer creates GET requests in regular intervals. Depending on the selected sonifications strategy, the movement values are then mapped to control parameters of the synthesizer, eventually creating the perceived sound. For the creation of rhythmic patterns a simple web based drum machine has been implemented. A set of four pattern generators is used. These pattern generators function like a set of gears (see [9]), each producing a configurable repetitive rhythmic pattern. The gears could also be linked, thus being synchronized. For each gear up to two independent sounds could be selected with a pattern length between 1 and 17 beats for a single gear revolution. The sounds of the drum machine are produced by the described web audio synthesizer.

<sup>1</sup> A secondary simple sonification system has been implemented using SuperCollider. Here parameters are transmitted using OSC.

### 3.1 Adaptive Sonification

The processing proxy performs the central analytic part of the current systems. Basically it reduces the vast amount of sensor readings to a set of relevant control parameters.

This transformation process is also parameterized and can thus be dynamically adapted to body movements. This is achieved by integrating the wekinator (see [10]). The wekinator is connected via Open Sound Control (OSC) and is able to learn a mapping function for given parameters.

In order to create a dynamic mapping, fixed values of control parameters are paired with certain postures of the human body. After presenting the system just a couple of relevant examples, the wekinator is able to create an interpolation model for this parameter.

This approach makes it relatively simple to adapt to the specific movement abilities of a system's user.

## 4. FIRST EXPERIMENTS

A set of different sonification strategies has been implemented. Parameters on a number of musical levels and sound levels are controlled by the preprocessed movement measurements.

- pitch
- volume
- rhythm
- complexity of tonal clusters (filter parameters of noise)
- chord selection
- melody structure
- position in the stereo field
- position in a spatial field

The simplest way to sonify value changes is a direct mapping from movement to pitch. In our case, we experimented with speed and acceleration of the arm movement. While speed leads to smaller, less abrupt changes in pitch, acceleration creates a sound impression, that can be compared to a theremin. We also tried out to simultaneously sonify different sensors with different sounds (and also positioned them in the stereo field). While the sound became more interesting in a musical sense, the bio feedback seemed to be too complex. It became quite complicated to map the perceived sound back to the movement control of the arm. The second experiment tried to keep the pitch of the sonification steady. This allowed for a better control of the harmonic structure of the auditory feedback. Instead the volume of the sound was mapped to the arm movements. It turned out, that a better speed control of the arm movement could be realized by the participants. On the other hand, it became relatively complicated to get an adequate feedback for fine grained arm movements. The differences in auditory feedback were barely noticeable, even though, the volume mapping used a logarithmic scale. In our experiment on rhythmic structure we followed ideas

inspired by Toussaint ([9]). A rhythm machine was implemented by means of geometrical descriptions. Here we (conceptually) used gears of different sizes to create rhythmic patterns of different length and speed. The arm movements were mapped to these parameters, thus changing the overall speed of the rhythm, as well its internal structure. Participants liked this kind of feedback. It gave them a good control about the temporal course of their movements. Even slight variations of movement speed were easily detected. On the other hand, variations in the movement measurement needed to be smoothed out more. For the next experiment we tried to provide as little musical structure as possible. Instead, multi band filtered pink, white and brown noise was used to create a non disturbing pleasant background hiss, comparable to an ocean noise on the seaside. The filter cutoff frequencies were modulated by a set of slowly moving LFOs, which in turn were controlled by the arm movement data. Here the acoustic feedback was perceived as delayed, not directly connected to the arm movement. Nevertheless, participants kind of liked it, as it provided a means of a slow moody change. They realized something went wrong a little time ago, moved back to a previous position and repeated the movement, hoping to get no further negative acoustic feedback.

The experiment on chord selection was based on a variation of a Tonnetz by Euler. Instead of moving through the network in a circular way, a given central chord was chosen, and the arm movements were mapped to a distance value, thus moving away from the central chord to select more distant chords. Once the arm movement was back on track, closer chords were chosen again. Most of the participants liked to stay within a close range to the central chord, thus mainly producing simple (musically rather dull) cadences of tonica, dominante and sub-dominante. Still some also enjoyed the more complex structure of the distant chords. The perceived musical structure led to a stronger distraction of the participants, as they rather tried to get back to the central chord, than concentrating onto the correct arm movement. In a sense, the participants used the sonification system for a musical performance. The final three sonification strategies have not yet been under experimental testing. If accepted to the conference, we hope to be able to present all of our results as part of a poster/demo presentation.

## 5. CONCLUSIONS

In this paper we present the early stage of our research on the development of a web audio based sonification system for body movement data. So far, we have been able to design and implement a first version of the system. This prototype is fully based on current web technology. We developed a set of sonification strategies and conducted a number of experiments to pre test our hypotheses. The results are quite promising. Within the context of physiotherapy we expect to achieve positive effects on the rehabilitation of patients by integrating auditory bio feedback into the therapy. In a second strand of research, we would like to use the system in a more musical way. More specifically, we would like to use the system to capture and analyse the

arm and body movements of a violist (our second co author) and use the incoming body movement data to control the sonification system in a musical and esthetic way. As the violist is also teaching viola playing to younger children, one might also use our system in a didactic setting.

## 6. REFERENCES

- [1] R. Bloom, A. Przekop, and T. D. Sanger, "Prolonged electromyogram biofeedback improves upper extremity function in children with cerebral palsy," *Journal of child neurology*, vol. 25, no. 12, pp. 1480–1484, 2010.
- [2] G. W. Cronin and R. L. Steenerson, "The effectiveness of neuromuscular facial retraining combined with electromyography in facial paralysis rehabilitation," *OtolaryngologyHead and Neck Surgery*, vol. 128, no. 4, pp. 534–538, 2003.
- [3] J. Crow, N. Lincoln, F. Nouri, and W. d. Weerdt, "The effectiveness of emg biofeedback in the treatment of arm function after stroke," *International disability studies*, vol. 11, no. 4, pp. 155–160, 1989.
- [4] C. Dohle, N. Morkisch, R. Lommack, and L. Kadow, "Spiegeltherapie," *neuoreha*, vol. 3, no. 04, pp. 184–190, 2011.
- [5] M. Dozza, L. Chiari, and F. B. Horak, "Audio-biofeedback improves balance in patients with bilateral vestibular loss," *Archives of physical medicine and rehabilitation*, vol. 86, no. 7, pp. 1401–1403, 2005.
- [6] H.-Y. Huang, J.-J. Lin, Y. L. Guo, W. T.-J. Wang, and Y.-J. Chen, "Emg biofeedback effectiveness to alter muscle activity pattern and scapular kinematics in subjects with and without shoulder impingement," *Journal of electromyography and kinesiology*, vol. 23, no. 1, pp. 267–274, 2013.
- [7] S. Seidel, G. Kasprian, T. Sycha, and E. Auff, "Spiegeltherapie bei phantomschmerzen," *Wiener klinische Wochenschrift*, vol. 121, no. 13-14, pp. 440–444, 2009.
- [8] E. Dalla Toffola, C. Tinelli, A. Lozza, M. Bejor, C. Pavese, I. Degli Agosti, and L. Petrucci, "Choosing the best rehabilitation treatment for bells palsy," *Eur J Phys Rehabil Med*, vol. 48, no. 4, pp. 635–642, 2012.
- [9] G. T. Toussaint, *The Geometry of Musical Rhythm: What Makes a "Good" Rhythm Good?* Chapman and Hall/CRC, 2016.
- [10] M. Schedel and R. Fiebrink, "A demonstration of bow articulation recognition with wekinator and k-bow," in *ICMC*, 2011.

# VOCALISTMIRROR: A SINGER SUPPORT INTERFACE FOR AVOIDING UNDESIRABLE FACIAL EXPRESSIONS

Kin Wah Edward Lin, Tomoyasu Nakano, Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST)

Central 2, 1-1-1, Umezono, Tsukuba, Ibaraki 305-8568, Japan

{edward.lin, t.nakano, m.goto}@aist.go.jp

## ABSTRACT

We present *VocalistMirror*, an interactive user interface that enables a singer to avoid their undesirable facial expressions in singing video recordings. Since singers usually focus on singing expressions and do not care about facial expressions, when watching singing videos they recorded, they sometimes notice that some of their facial expressions are undesirable. *VocalistMirror* allows a singer to first specify their undesirable facial expressions in a recorded video, and then sing again while seeing a real-time warning that is shown when the facial expression of the singer becomes similar to one of the specified undesirable expressions. It also displays Karaoke-style lyrics with piano-roll melody and visualizes acoustic features of singing voices. iOS ARKit framework is used to quantify the facial expression as a 52-dimensional vector, which is then used to compute the distance from undesirable expressions. Our experimental results showed the potential of the proposed interface.

## 1. INTRODUCTION

Although many tools to enable singers to achieve desirable singing expressions of their singing voices have been developed [1–4], to the best of our knowledge, no tools have been developed to enable singers to avoid undesirable facial expressions while singing. For example, Lin et al. [5] developed a singing pitch training interface that visualizes singing pitch (fundamental frequency of singing voice) in real time and gives feedback on the correctness of the pitch while singing. Tsuzuki et al. [6] developed an interface to create derivative choruses by mixing (mashing up) various singing voices sung by different singers for the same song. Such a variety of singing voices are typically available as vocal covers on video sharing services. Ojima et al. [7] developed a real-time vocal-part arrangement system that enables a user to manipulate the vocal part of existing music audio signals. Fragments of singing voices can be manipulated and played back by using a MIDI keyboard. These interfaces and systems focused on singing voices and did not deal with facial expressions in singing.

The importance of facial expressions has already been investigated in the literature. Quinto et al. [8] stated that facial expressions of a singer are known to influence the perception of singing expressiveness, tension, timbre, dissonance, note duration, interval size, phrase structure, and emotion. They also reported the relationship between the singing emotion and the pre-production, production, and post-production of facial expressions. Lyons and Tetsutani [9] and Koh and Yadegari [10] demonstrated how facial expressions can be used to manipulate audio signals. Goto et al. [11] showed how singing and facial expressions of a singer can be imitated by a humanoid robot. However, there are no studies on applications helping singers have desirable facial expressions while singing.

The popularity of recording short singing video clips and uploading them to social media has increased. This can be considered a new form of music interaction and this trend is the most observable among young people. Its popularity is evident in the large market size of singing apps (i.e., smartphone/tablet applications for singing) that enable users to create singing video recordings. For example, TikTok<sup>1</sup>, Smule<sup>2</sup>, and Yokee<sup>3</sup> are popular and provide functions to add digital makeup or decorate faces and backgrounds. Those makeup functions can be manually used by singers at the post-production stage, but singing apps are not able to detect undesirable facial expressions of singers.

We therefore propose a singer support interface called *VocalistMirror* that enables a user to record a short singing video clip with desirable facial expressions, which could be uploaded to social media by the singer. With the *VocalistMirror* interface, the user can sing an excerpt of a song while listening to its karaoke track and seeing automatically-scrolling Karaoke-style lyrics with a piano-roll melody line. Acoustic features of the user's singing voice such as the sung pitch (fundamental frequency), and timing are analyzed and visualized in real time. This visualized feedback helps the user be aware of the accuracy of the sung pitch and timing, and thus helps users improve their singing expressions. Furthermore, the user's singing voice and facial expressions are automatically recorded as a video clip and played back after singing. During the

Copyright: © 2019 Kin Wah Edward Lin, Tomoyasu Nakano, Masataka Goto This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup> TikTok by Bytedance <https://itunes.apple.com/us/app/id835599320> accessed on: 15 Feb 2019.

<sup>2</sup> Smule - The #1 Singing App by Smule <https://itunes.apple.com/us/app/id509993510> accessed on: 15 Feb 2019.

<sup>3</sup> Karaoke - Sing Unlimited Songs by Yokee Music <https://itunes.apple.com/us/app/id547109049> accessed on: 15 Feb 2019.



playback of this singing video clip, VocalistMirror allows the user to select several frames of *undesirable* facial expressions that the user sometimes wears. Since the user typically does not notice those undesirable facial expressions while singing, the user wants to avoid them. VocalistMirror solves this problem by letting the user sing again while automatically analyzing the user's facial expressions in real time and displaying a warning when one of the selected undesirable facial expressions is detected. VocalistMirror thus acts as a *mirror* enabling the user to notice when the user wears an undesirable facial expression and avoid it in a recorded singing video clip.

Our main contributions are three-fold: (1) we opened up a new way of assisting singers from the viewpoint of facial expressions, (2) we designed and implemented an interface helping singers specify and avoid their undesirable facial expressions in a simple intuitive way, and (3) we evaluated the effectiveness and potential of the interface by conducting a user study.

## 2. SYSTEM DESIGN AND IMPLEMENTATION

We want our VocalistMirror to be a tool that is capable of creating a short singing video clip with a duration similar to that of the 15 to 30 seconds video clips posted on social media such as TikTok, so that it could encourage novice-level singers to record their own singing video clips. We also want VocalistMirror to be easily accessible and easy-to-use. In the following three subsections, we discuss what platform our VocalistMirror should be deployed on, how facial expressions are quantified, and what acoustic features should be visualized.

### 2.1 Deployment Platform

Among available popular platforms such as Windows, macOS, Android, and iOS, we decided to develop and deploy our interface on the iOS platform. It is portable and accessible given the size of mobile devices such as iPhone and iPad. It also provides strong software and hardware support for analyzing facial expressions and acoustic features. This choice means we can use Apple's TrueDepth camera system [12, 13] that is available on the iOS platform and expect the audio quality to be high enough<sup>4</sup>.

### 2.2 Quantification of Facial Expressions

The iOS platform has a software framework called ARKit<sup>5</sup> that can quantify facial expressions. It uses a front-facing TrueDepth camera on the iOS device to provide real-time analysis of singer's facial expressions. ARKit quantifies each facial expression as a facial wireframe with 1,220 vertices. It can further analyze those vertices to provide 52 distinct facial shapes<sup>6</sup>, which are classified into 5 categories: (1) Left Eye, (2) Right Eye,

<sup>4</sup> Mobile Audio Quality Index from JUCE <https://juce.com/maq> accessed on: 15 Feb 2019.

<sup>5</sup> ARKit <https://developer.apple.com/documentation/arkit> accessed on: 15 Feb 2019.

<sup>6</sup> ARKit Face Blendshape by Apple <https://developer.apple.com/documentation/arkit/arfaceanchor/blendshapelocation> accessed on: 15 Feb 2019.

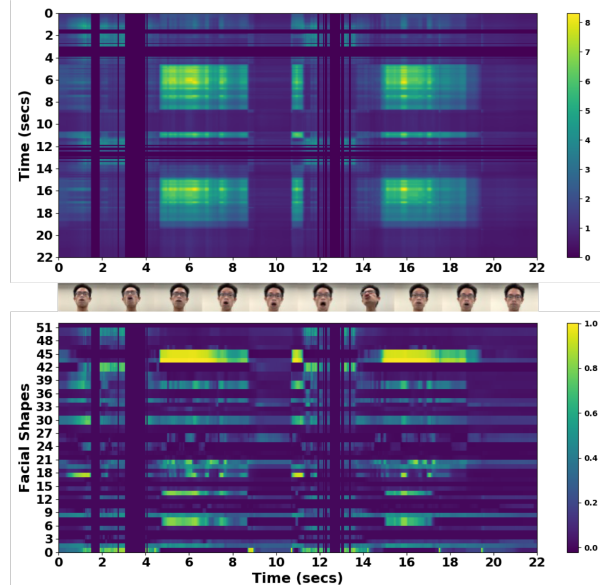


Figure 1. Feasibility study using ARKit framework to quantify facial expressions.

(3) Mouth and Jaw, (4) Eyebrows, Cheeks, and Nose, and (5) Tongue. Since each of the 52 facial shapes has a value ranging from 0 to 1, we can regard a set of values of the 52 facial shapes as a 52-dimensional *facial vector* that represents the facial expression in a video frame. We quantify how similar the facial expressions in two different video frames are by calculating the distance between the two vectors of those frames.

To illustrate the feasibility of using these 52-dimensional facial vectors to represent and detect similar facial expressions, we first asked a singer to sing a 22-second song that repeats two similar musical phrases. We also asked the singer to try to express the same facial expression for each of the musical phrases. We used the ARKit framework to capture a set of facial vectors during this singing performance. Then we calculated a self-similarity matrix of this set of vectors. Fig. 1 shows the self-similarity matrix of these facial vectors on the top, snapshots of singer's facial expressions in the middle, and the corresponding facial vectors at the bottom. This figure clearly shows that similar facial expressions of similar phrases were repeated twice. We can therefore use the 52-dimensional facial vectors obtained from the ARKit framework to detect similar facial expressions.

The similarity between facial expressions is quantified by calculating the exponential moving average of the L1-norm distance. The smaller the L1-norm distance is, the more similar they are. The real-time warning for undesirable facial expressions is displayed only when the L1-norm distance is below a threshold. We leave other distance calculations and the corresponding threshold setting for future work.

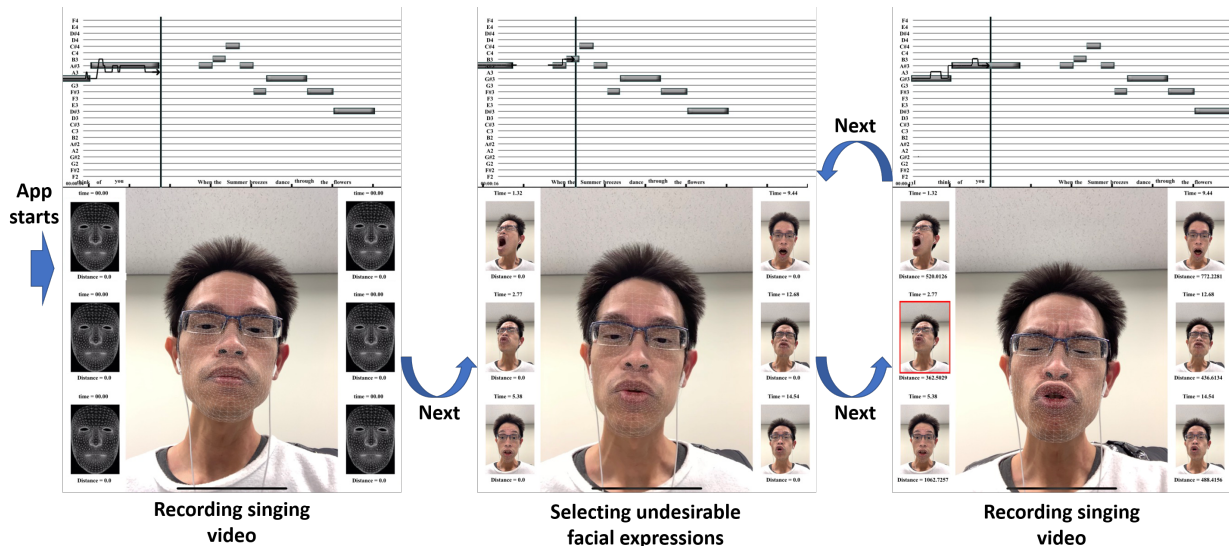


Figure 2. Interface and workflow of VocalistMirror.

### 2.3 Visualization of Acoustic Features

To sing better, singers have to intentionally control several audio features of their singing voices, including but not limited to, the singing pitch (fundamental frequency), timing, vibrato, tremolo, and breathing. Among these acoustic features, we chose the singing pitch, timing, and vibrato for the visualization, and we leave the others for future work. To visualize the singing pitch and vibrato in the right timing in real time, we need a fast and accurate fundamental frequency estimator. A third-party iOS AudioKit framework<sup>7</sup> not only provides such an estimator [14], but also seamlessly cooperates with the iOS platform that we use for the ARKit framework. Since AudioKit also provides various audio effects and analysis tools, VocalistMirror could use them to extend its functions in the future. For example, we could use the facial expressions to call the AudioKit API to create the Wah-Wah audio effect on the singing voice. We could also use AudioKit analysis tools to visualize the singing formants so that the phonetic quality of the singing voice could be visually examined.

## 3. INTERFACE DESIGN

With the above design principles in mind, we implemented and deployed the VocalistMirror application on an 11-inch iPad Pro 2018 with iOS 12.1.4. Since Lin [15] showed that many singers prefer a larger screen for singing apps, we chose iPad, rather than iPhone for our implementation. In this section, we first describe the workflow of VocalistMirror and then discuss its interface details.

### 3.1 Workflow of VocalistMirror

Fig. 2 shows an overview of our proposed interface. We call the lower part of the interface a *facial-expression interface* and the upper part of the interface an *acoustic-feature*

*interface*.

When the VocalistMirror app starts, it is at the stage of singing video recording, which is shown in the left of Fig. 2. A singer is expected to sing a short excerpt of a specified song while looking at the acoustic-feature interface as a singing guide. While the singer is singing and listening to the corresponding karaoke accompaniment, the acoustic-feature interface uses the AudioKit framework to provide real-time visual feedback on the singing pitch (fundamental frequency), timing, and vibrato. The singing voice and its corresponding facial expressions are recorded separately into two different files, one for the audio track and the other for the video track. After the singer finishes recording, VocalistMirror is moved to the next stage where the singer can play back and watch the recorded video clip.

At the second stage of selecting undesirable facial expressions, which is shown in the center of Fig. 2, the acoustic-feature interface uses the AudioKit framework to trace the fundamental frequency of the recorded singing voice in the audio track, so that the singer is able to see the pitch, timing and vibrato of the recorded singing voice. Moreover, while the facial-expression interface plays back the recorded video track, the singer is expected to select the singer's undesirable facial expressions by tapping the iPad screen. Due to the limitation of the screen size, the singer can select only six or fewer undesirable facial expressions. The snapshot of each selected facial expression is displayed along with its occurrence time in the video clip. Once the singer finishes selecting undesirable facial expressions, VocalistMirror is moved back to the stage of singing video recording, which is shown in the right of Fig. 2.

The singer at this time is expected to record a new and more satisfying singing video clip by avoiding the selected undesirable facial expressions. The facial-expression interface helps the singer do so by displays a real-time warning when the singer's current facial expression in singing

<sup>7</sup> AudioKit <https://audiokit.io/> accessed on: 15 Feb 2019.

becomes very similar to one of the selected undesirable expressions. This warning is represented by adding a red rectangular border to the snapshot of the most similar expression.

These two stages keep alternating until the singer selects another song, quits the app, or most likely, is satisfied with a recorded short singing video clip without undesirable facial expressions

### 3.2 Facial-Expression Interface Design

The facial-expression interface is used to record facial expressions of the singing performance and display the similarity between the singer's current facial expression and each of the selected undesirable facial expressions. To make sure that the facial expression is well captured by the ARVideoKit framework<sup>8</sup>, a thin facial wireframe is overlaid on the singer's face whenever the singer faces the TrueDepth camera. Therefore, when the thin facial wireframe disappears, the disappearance alerts the singer to face the TrueDepth camera.

### 3.3 Acoustic-Feature Interface Design

Since the acoustic-feature interface is mainly used as the singing guide, the singer is likely to concentrate on this interface while singing. And since the TrueDepth camera is located at the middle top of the iPad screen, we position this interface at the upper part of the screen so that the singer is more likely to face the camera. The display of the piano-roll melody line as well as the karaoke-style lyrics is implemented by using the Songle web service [16] for the melody line and the TextAlive web service [17] for the lyrics timing. We use SpriteKit<sup>9</sup> to implement the graph drawing. The singing pitch, timing, and vibrato are visualized using the arrow on the timeline cursor shown in Fig. 2.

## 4. EXPERIMENT

In this section, we first describe the setup of our subjective experiment. We then report and discuss experimental results. This preliminary experiment illustrates the effectiveness and potential of the proposed interface.

### 4.1 Experiment Setup

Each participant used VocalistMirror alone in a separated and quiet room to create their recording of an excerpt (the first two musical phrases) from *RWC-MDB-P-2001 No.87* in the RWC Music Database (Popular Music) [18]. The duration of the excerpt is 22 seconds. Since each participant was also told to use earphones to listen to the karaoke accompaniment of this song, accompaniment sounds were not recorded by the microphone the participant used and we could record solo singing without any accompaniment. The experiment followed the steps described below and on average took half an hour for each participant. After the

experiment, it was optional for the participant to provide further feedback and suggestions.

#### 4.1.1 Pre-experiment questionnaires:

Each participant first filled out an online form that asked about their age (like 20's, 30's, or 40's), gender, musical background, and experience creating their own singing video clips.

#### 4.1.2 Introduction and demonstration:

We then used about 10 minutes to explain and demonstrate the intended usage of VocalistMirror. During this tutorial, we also clarified whatever questions they raised so that they would be comfortable using VocalistMirror alone in a room to create a singing video clip.

#### 4.1.3 Singing video recording:

Since we asked the participants to use VocalistMirror until they had finished a satisfying recording, each participant knew that there was no time limit on recording a video clip.

#### 4.1.4 Post-experiment questionnaires:

Once they finished their recording, they filled out another online form alone in the room. The form was used to evaluate VocalistMirror on a 7-point scale of degree of appreciation, with 7 being the most and 1 being the least. They were told that they should answer the following questions with their first impressions.

- How much do you like the exterior design of the app? (i.e., The graphics and the user interface.)
- How much do you like the features of avoiding undesirable facial expressions?
- How much do you like the features of visualizing acoustic features of your singing voice?
- How much do you think the app has helped you improve your facial expressions after multiple uses?
- How much do you like the app as a whole?

### 4.2 Experimental Results

Eight participants (four male and four female) participated in this experiment. Their ages ranged from the 20's to the 40's. Five of them had basic western music education and had several years of musical instrument (e.g., piano) practices during their teenage period. These participants are considered to have had no serious music background. The other three had more advanced music education and had serious musical instrument practice (including singing) for at least six years. These participants are considered to have had a serious music background. Only one participant, who was the youngest, has used Instagram<sup>10</sup> to create a short singing video clip before. We conclude that these participants were suitable for evaluating our proposed interface because their genders were evenly distributed, they

<sup>8</sup> ARVideoKit <https://github.com/AFathi/ARVideoKit> accessed on: 15 Feb 2019.

<sup>9</sup> SpriteKit <https://developer.apple.com/documentation/spritekit> accessed on: 15 Feb 2019.

<sup>10</sup> Instagram <https://itunes.apple.com/us/app/id389801252> accessed on: 15 Feb 2019.

How much do you like	Least							Most		Med- ian	Mo- de
	1	2	3	4	5	6	7				
Exterior Design		2	1	2	3					4	5
Features of Avoiding		1	1	1	2	2	1			5	5,6
Features of Visualizing				4	3		1			5	4
Improving Expressions		2		2	1	2	1			4.5	2,4,6
Overall Impression			1	1	5		1			5	5

Table 1. Evaluation of the proposed interface by all eight participants. Each number represents the number of participants who selected the corresponding item.

How much do you like	Least							Most		Med- ian	Mo- de
	1	2	3	4	5	6	7				
Exterior Design		2	1							2	2
Features of Avoiding		1	1		1					3	2,3,5
Features of Visualizing				3						4	4
Improving Expressions		2		1						2	2
Overall Impression			1	1	1					4	3,4,5

Table 2. Evaluation of the proposed interface by three participants who had a serious music background. Each number represents the number of participants who selected the corresponding item.

How much do you like	Least							Most		Med- ian	Mo- de
	1	2	3	4	5	6	7				
Exterior Design				2	3					5	5
Features of Avoiding				1	1	2	1			6	6
Features of Visualizing				1	3		1			6	6
Improving Expressions				1	1	2	1			6	6
Overall Impression					4		1			5	5

Table 3. Evaluation of the proposed interface by five participants who had no serious music background. Each number represents the number of participants who selected the corresponding item.

were from different generations, and their music backgrounds were almost evenly distributed.

Table 1 shows their responses toward the post-experiment questionnaires. Based on the median values, participants had positive impressions (4 and above) towards VocalistMirror. The lowest median value (4) in the exterior design suggests that we should invite a professional graphic designer to improve our interface in terms of aesthetics. By identifying the participants who mostly gave points above or below 4, we realize there could be two distinct user groups.

Table 2 shows the responses of the three participants who had a serious music background. Table 3 shows the responses of the other five participants, who had no serious music background. By comparing the responses of these two groups of people, we realize that each group of people has distinct and contradictory opinions of VocalistMirror. Examples are shown below.

- Participants with no serious music background would appreciate more on the features of avoiding undesirable facial expressions and would more agree that their facial expression is improved after multiple uses. They mentioned that the alternating stage design makes VocalistMirror easy to use and, most importantly, encourages them to be more aware of their undesirable facial expressions. They felt that a process of simply selecting some undesirable facial expressions is sometimes helpful enough for them to avoid undesirable facial expressions. However, the other group of participants with a serious music background less agreed as they would demand more sophisticated features for selecting their undesirable facial expressions in the first place. For example, they may only dislike the mouth shape or the head tilt. Hence, they demand a precise navigation function so they can navigate the recorded video clip precisely to find that particular set of frames.

- Participants with serious music background would demand the interface design to match their music understanding. For example, once the experienced singers know the key of the song after hearing the first few notes of the song, they just need the pitch information which is related to the key of the song (i.e., the solfeggio - do, re, me fa, so, la, si, do) in order to sing in tune. Hence, the experienced singers demand the interface to show the relative pitch information. However, novice-level singers would prefer the interface to display the absolute pitch information (e.g., C4, D4, and so on) so that they could feel the pitch control is just like pressing the note on the piano keyboard.

These opinions suggest that we will need two different versions of VocalistMirror in the future if we want to target both of the user groups. It also suggests that the current version of VocalistMirror is more appreciated by the participants with no serious music background.

## 5. CONCLUSION

We described *VocalistMirror*, a real-time user interface helping a singer avoid the singer's undesirable facial expressions. Although facial expression is an essential feature in singing, to the best of our knowledge, this is the first study that focuses on undesirable facial expressions of singers. VocalistMirror analyzes the singing voice taken from a microphone input in real time and visualizes its pitch trajectory as well as a piano-roll melody line with Karaoke-style lyrics scrolling automatically. Moreover, it analyzes the singer's facial expression taken from a camera in real time and displays a warning if it becomes similar to one of undesirable facial expressions specified by the singer. In our current implementation, VocalistMirror uses iOS ARKit framework to quantify the facial expression as a 52-dimensional facial vector with each dimension ranging from 0 to 1. We can tell how similar two facial



expressions are by calculating the exponential moving average of the L1-norm distance between the corresponding two facial vectors. Experimental results showed the effectiveness and potential of the proposed interface and, most importantly, they provide a direction for further improving our VocalistMirror interface.

### Acknowledgments

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, Japan. In this work, we used the RWC Music Database (RWC-MDB-P-2001) [18].

### References

- [1] D. Hoppe, M. Sadakata, and P. Desain, “Development of real-time visual feedback assistance in singing training: A review,” *Journal of Computer Assisted Learning*, vol. 22, pp. 308–316, 2006.
- [2] F. Moschos, A. Georgaki, and G. Kouroupetroglou, “FONASKEIN: An interactive application software for the practice of the singing voice,” in *Proc. Sound and Music Computing Conference (SMC 2016)*, 2016, pp. 326–331.
- [3] R. Gong, Y. Yang, and X. Serra, “Pitch contour segmentation for computer-aided jingju singing training,” in *Proc. Sound and Music Computing Conference (SMC 2016)*, 2016, pp. 172–178.
- [4] M. Pérez-Gil, J. Tejada, R. Morant, D. Martos, and A. Pérez-González, “Cantus: Construction and evaluation of a software solution for real-time vocal music training and musical intonation assessment,” *Journal of Music, Technology & Education*, vol. 9, no. 2, pp. 125–144, 2016.
- [5] K. W. E. Lin, H. Anderson, M. Hamzeen, and S. Lui, “Implementation and evaluation of real-time interactive user interface design in self-learning singing pitch training apps,” in *Proc. International Computer Music Conference and Sound and Music Computing Conference (Joint ICMC SMC 2014 Conference)*, 2014, pp. 1693–1697.
- [6] K. Tsuzuki, T. Nakano, M. Goto, T. Yamada, and S. Makino, “Unisoner: An interactive interface for derivative chorus creation from various singing voices on the web,” in *Proc. International Computer Music Conference and Sound and Music Computing Conference (Joint ICMC SMC 2014 Conference)*, 2014, pp. 790–797.
- [7] Y. Ojima, T. Nakano, S. Fukayama, J. Kato, M. Goto, K. Itoyama, and K. Yoshii, “A singing instrument for real-time vocal-part arrangement of music audio signals,” in *Proc. Sound and Music Computing Conference (SMC 2017)*, 2017, pp. 443–449.
- [8] L. R. Quinto, W. F. Thompson, C. Kroos, and C. Palmer, “Singing emotionally: a study of pre-production, production, and post-production facial expressions,” *Frontiers in Psychology*, vol. 5, p. 262, 2014.
- [9] M. J. Lyons and N. Tetsutani, “Facing the music: A facial action controlled musical interface,” in *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, 2001, pp. 309–310.
- [10] E. S. Koh and S. Yadegari, “Mugeetion: Musical interface using facial gesture and emotion,” in *Proc. International Computer Music Conference (ICMC 2018)*, 2018.
- [11] M. Goto, T. Nakano, S. Kajita, Y. Matsusaka, S. Nakaoka, and K. Yokoi, “VocaListener and VocaWatcher: Imitating a human singer by using signal processing,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (IEEE ICASSP 2012)*, 2012, pp. 5393–5396.
- [12] M. Abbaszadegan, S. Yaghoubi, and I. S. MacKenzie, “Trackmaze: A comparison of head-tracking, eye-tracking, and tilt as input methods for mobile games,” in *Proc. HCI International 2018*, 2018, pp. 393–405.
- [13] A. Scicali, I. Nwogu, and J. Geigel, “Mobile facial emotion recognition engine,” in *ACM Symposium on Applied Perception*, 2018.
- [14] J. C. Brown and M. S. Puckette, “A high resolution fundamental frequency determination based on phase changes of the Fourier transform,” *The Journal of the Acoustical Society of America (JASA)*, vol. 94, no. 2, pp. 662–667, 1993.
- [15] K. W. E. Lin, “Singing voice analysis in popular music using machine learning approaches,” Ph.D. dissertation, Singapore University of Technology and Design, 2018.
- [16] M. Goto, K. Yoshii, H. Fujihara, M. Mauch, and T. Nakano, “Songle: A web service for active music listening improved by user contributions,” in *Proc. International Society for Music Information Retrieval Conference (ISMIR 2011)*, 2011, pp. 287–288.
- [17] J. Kato, T. Nakano, and M. Goto, “TextAlive: Integrated design environment for kinetic typography,” in *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (ACM CHI 2015)*, 2015, pp. 3403–3412.
- [18] M. Goto, T. Nishimura, H. Hashiguchi, and R. Oka, “RWC Music Database: Popular, classical, and jazz music databases,” in *Proc. International Conference on Music Information Retrieval (ISMIR 2002)*, 2002, pp. 287–288.

# AUDIOVISUAL PERCEPTION OF AROUSAL, VALENCE, AND EFFORT IN CONTEMPORARY CELLO PERFORMANCE

Hanna Järveläinen

ICST, Zurich University of the Arts  
hanna.jarvelainen@zhdk.ch

## ABSTRACT

Perceived arousal, valence, and effort were measured continuously from auditory, visual, and audiovisual cues using a recorded performance of a contemporary cello piece. Effort (perceived exertion of the performer) was added for two motivations: to investigate its potential as a measure and its association with arousal in audiovisual perception. Fifty-two subjects participated in the experiment. Results were analyzed using Activity Analysis and functional data analysis. Arousal and effort were perceived with significant coordination between participants from auditory, visual, as well as audiovisual cues. Significant differences were detected between auditory and visual channels but not between arousal and effort. Valence, in contrast, showed no significant coordination between participants. Relative importance of the visual channel is discussed.

## 1. INTRODUCTION

Describing emotional response to music, the circumplex model presents a variety of affects as a combination of two dimensions, arousal and valence (see [1, 2] for a review). In tonal classical music, the main cues for high arousal are fast tempo and high intensity, while major mode, fast tempo, and high pitch contribute to positive valence [3]. Western listeners learn cultural valence cues from classic-romantic and from popular music. These cues are often absent in contemporary music, where valence must be judged from acoustic attributes. Valence models are less successful under such circumstances [4]. Dean and Bailes found a potential association between valence and spectral flatness in electroacoustic music [5]. However, the effect was also overruled by higher-level features.

Dean and Bailes also discovered loudness patterns in electronic compositions resembling those resulting from the player's exertion in classical music [6]. They suggested that effort would be a key element in the FEELA-chain leading to emotional arousal (Force, Energy, Effort, Loudness, Arousal). In the auditory domain, effort is associated with intensity, source complexity, and event density in varying degrees, as long as human musical agency is apparent to the listener [7]. In the visual domain, effort is

considered a factor of expressive body movement in musical and dance performance. However, perception of effort changes has been less explored.

In music performance, various visual cues have been identified from gestures communicating specific affects such as anger, fear, grief, and joy [8]. General importance of visual kinematic cues has been confirmed for higher-level features such as performance judgment [9], expertise [10], player identification [11], structural phrasing [12, 13], expressive intentions [14–16], and various emotional cues [17–19]. However, basic affect perception from audiovisual cues has received less attention until recently. Vines and colleagues measured continuous perception of musical tension, reporting great differences between the auditory and visual channels [12]. Vuoskoski et al. studied expressivity [20] and emotional impact [21], concluding that visual cues were of equal or even higher importance than auditory cues. Yet the relative importance of visual cues in real-time perception still requires research.

The present experiment is motivated through the recent evidence of the role of the visual channel. Effort was included in order to explore its suitability as a measure in contemporary repertoire and its differences with arousal in audiovisual perception. The cello was chosen for good visibility of the playing gestures and their importance to loudness control. Based on literature and on pilot experiments, it is expected that both arousal and effort will be judged reliably from auditory and from visual cues. The two measures are expected to be positively associated. Suitability of valence as a measure of perceived affect in contemporary repertoire is discussed.

## 2. EXPERIMENT

### 2.1 Design, stimuli, and apparatus

Three factors were varied in the experiment: measurement (arousal, valence, and effort), sensory modality (A=auditory only, V=visual only, and AV=audiovisual), and musical material (segments 1-3). The material consisted of three excerpts from an audio-video recording of the solo cello work *Pression* by Helmut Lachenmann. This piece was chosen because it lacks melodic and harmonic elements as well as regular beats, yet it contains a rich variety of playing techniques, gestures, and timbres. The video recording was made using a professional camera. The performer was filmed from a distance of ca four meters against a still, dark, and neutral background, offering good contrast to her

Copyright: © 2019 Hanna Järveläinen et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

clothing. The audio track was two-channel at 48 kHz.<sup>1</sup> The duration of the segments varied between 2 min and 2 min 30 sec. The unimodal A and V conditions contained only the audio and the video tracks, respectively, while the bimodal AV condition contained both. Audio and video were always congruent.

Subjects made continuous ratings of perceived (not experienced) affect using a slider while observing the performance one factor combination at a time. A custom-made software played back the excerpts on a 13-inch laptop computer and collected responses at a sampling frequency of 4 Hz. Audio was presented through Direct Sound EX-29 Extreme Isolation headphones at realistic and comfortable level, and video was shown in full-screen mode. Subjects were instructed to rate perceived, not experienced affect, and the meanings of the three measurements were explained to them before each measurement block. Arousal was described varying between tense/relaxed or awake/tired. Effort was defined as the musician's exertion in order to produce the sound. For these measures, the slider setting was mapped to the numeric range [0,1]. Valence was described as varying between positive/negative, pleasant/unpleasant, attractive/unattractive, or happy/sad. The slider was mapped to the range [-0.5, 0.5].

## 2.2 Subjects and procedure

Fifty-two students participated in the experiment (ages 20-45 years,  $M = 27$ ; 16 M, 36 F). Roughly two thirds were music majors, the rest had no significant musical experience. Participants' musical background was recorded but not controlled as a grouping factor. The session took ca 30 minutes. After completing the experiment, participants reported their liking and perceived familiarity of the repertoire, both on a scale from one to five.

Each participant was assigned nine of the 27 factor combinations as follows. At first, the six permutations of segment numbers 1-3 were assigned to the modalities A, AV, and V, producing triplets such as (A1, AV2, V3). Then, one of these triplets was assigned to each measurement such that the participants received each modality once within the three segments of a single measurement, and each modality once within the three measurements of a single segment. There were 12 such sets; thus each set was received by four or five participants. An example of the conditions received by a single subject is given in Table 1.

This scheme produced 17 ratings per measurement-modality-segment bin (16-18 due to practicalities). There were no common participants in bins with matching modality and segment (for example, effort-A-1, arousal-A-1, valence-A-1), nor in bins with matching measurement and segment (for example, effort-A-1, effort-V-1, effort-AV-1). However, between non-matching factor combinations (such as effort-A and valence-AV), there was up to 50% overlap between participants. The three measurements were presented as blocks in balanced order.

	A	AV	V
Arousal	Seg. 1	Seg. 2	Seg. 3
Valence	Seg. 2	Seg. 3	Seg. 1
Effort	Seg. 3	Seg. 1	Seg. 2

Table 1: One of the 12 sets of factor combinations.

## 2.3 Data analysis

Coordination in participants' responses was investigated using Activity Analysis, a novel analytical framework based on alignment between continuous responses of different subjects [22]. Well aligned responses are probably driven by the stimulus and not produced randomly. Activity Analysis begins by searching individual responses for active events in terms of enough change in a given time frame. Activity levels are then computed as the proportion of responses that show a similar kind of event within a given time window of synchrony. Sequenced assessment of activity levels over the duration of the measurement produces the activity level time series. In this study, activity levels are computed from rating increases of at least 2.5% within 2-second time windows. This in turn is used for computing the Coordination Score by testing the distribution of activity levels against a parametric model of uncoordinated random activity. The single-number C-Score varies between 0 and 16, with  $C > 2$  indicating significant coordination on a  $p < 0.01$  level. A Bi-Coordination Score can be computed between two collections of ratings with different response conditions and participants. These analyses were performed in Matlab using the Activity Analysis toolbox [22].

Functional data analysis was performed in R using the *fda.usc* package [23]. For this analysis, the ratings were converted into functional data objects and then smoothed using nonparametric kernel estimation. The data were used in original as well as differenced form. To investigate differences in functional means between conditions, functional analysis of variance was performed based on randomly chosen one-dimensional projections [24]. Two-way between-subject ANOVAs were computed for each segment with measurement and modality as factors.<sup>2</sup>

## 3. RESULTS

### 3.1 Arousal and Effort

Functional means of combined arousal and effort ratings are presented in the top panels of Figures 2, 3, and 4<sup>3</sup>. Peak ratings are in all segments reached in the auditory condition. In segments one and three, peak auditory ratings top visual ratings by nearly 20 percentage points and audiovisual ratings by ca 10 pp. Crossmodal additive effects do not seem present; audiovisual ratings never exceed the higher unimodal condition.

<sup>2</sup> The segments were treated as separate experiments, as a comparison of different musical materials would not be meaningful.

<sup>3</sup> Averaging over the two measurements is justified by upcoming analysis.

<sup>1</sup> Recording available at <https://tube.switch.ch/videos/db27af24>

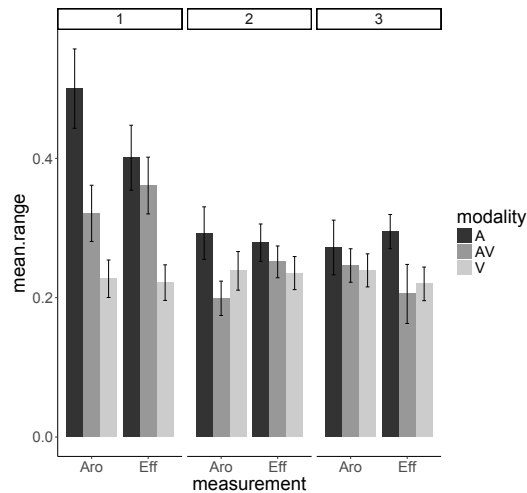


Figure 1: Mean ranges ( $\pm$ se) of the ratings curves in segments one, two, and three, as proportion of the ratings scale.

Participants' individual use of the rating scale was examined as the range between the 25% and 75% quantiles of their ratings. Figure 1 presents the mean ranges across participants. Typically, they used ca 25% of the rating scale, and the ranges were widest in the the auditory condition.

### 3.1.1 Activity Analysis

Coordination Scores for all factor combinations are given in Table 2. An example of ratings and the respective activity level time series is seen in Figure 5. In nearly all factor combinations, ratings were significantly coordinated ( $C > 2$ ), with an overall mean  $C = 3.79$ . However, only the auditory modality resulted in significant C-Scores in all factor combinations. The audiovisual modality was uncoordinated in two cases and the visual condition in one.

Bi-Coordination-Scores were computed to compare alignment between measurements and modalities (Table 3). The auditory and visual conditions were uncoordinated with only one exception. In contrast, auditory and audiovisual ratings were coordinated, except for arousal in segment one (mean Bi-C of the coordinated conditions = 3.66). Visual and audiovisual ratings were likewise coordinated, albeit with a lower mean Bi-C = 2.79. Arousal and effort measurements were always coordinated within matching modalities (mean Bi-C = 4.13).

The Activity Analysis results can be summarized as follows. Firstly, participants rated both arousal and effort in a coordinated way from isolated as well as combined auditory and visual cues. Secondly, there was significant bi-coordination between arousal and effort ratings in all modalities. Thirdly, even though there was generally no significant bi-coordination between auditory and visual ratings, both modalities were bi-coordinated with audiovisual ratings, suggesting that audiovisual perception is significantly driven by both auditory and visual cues.

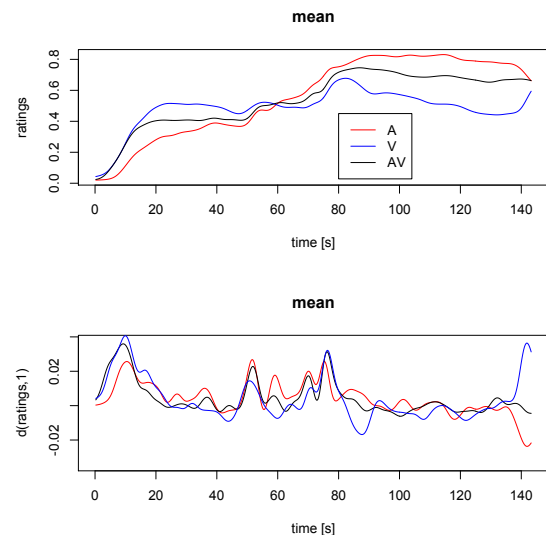


Figure 2: Original (top) and differenced (bottom) mean ratings (arousal and effort combined), segment one.

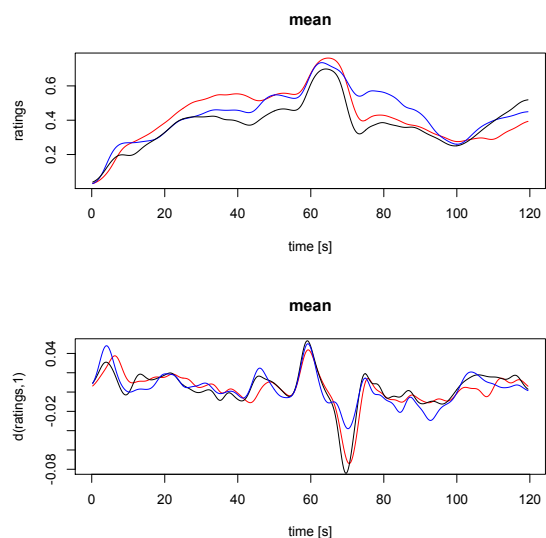


Figure 3: Original and differenced ratings, segment two.

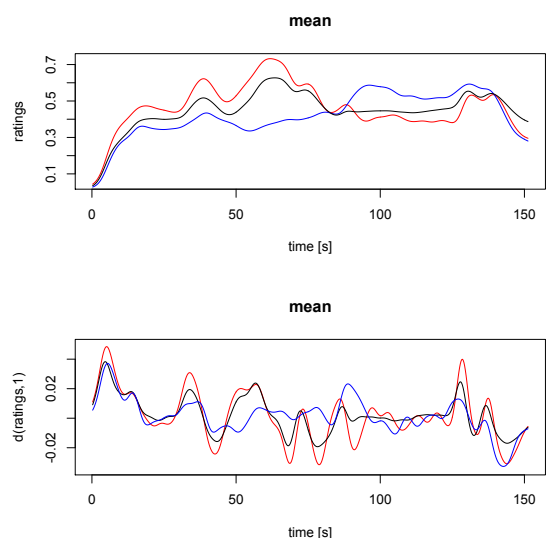


Figure 4: Original and differenced ratings, segment three.



	Arousal C-score	Effort C-score
Seg. 1	A: 3.36 V: 4.13 AV: < 2	A: 4.28 V: 2.80 AV: 4.40
Seg. 2	A: 3.06 V: 4.37 AV: 4.76	A: 4.71 V: < 2 AV: 4.76
Seg. 3	A: 2.70 V: 2.29 AV: 9.08	A: 2.42 V: 7.38 AV: < 2

Table 2: Activity Analysis C-Scores.

	Arousal Bi-C-score	Effort Bi-C-score	Aro-Eff Bi-C-score
Seg. 1	A-AV: < 2 V-AV: 2.25 A-V: < 2	A-AV: 2.94 V-AV: 2.25 A-V: < 2	A: 4.13 V: 3.40 AV: 3.68
Seg. 2	A-AV: 2.77 V-AV: 3.47 A-V: < 2	A-AV: 2.74 V-AV: 3.54 A-V: < 2	A: 3.10 V: 3.19 AV: 3.67
Seg. 3	A-AV: 4.67 V-AV: 2.90 A-V: 2.56	A-AV: 5.18 V-AV: 2.35 A-V: < 2	A: 5.65 V: 5.56 AV: 4.84

Table 3: Activity Analysis Bi-C-Scores.

### 3.1.2 Functional analysis of variance

Functional two-way ANOVA was computed segment-wise with measurement and modality as factors. A significant main effect was observed for modality: the p-value, obtained from 30 random projections, was  $p < 0.001$  for all three segments. On the contrary, the measurement effect was not significant in any segment ( $p \geq 0.43$ ), nor was the measurement:modality interaction ( $p \geq 0.16$ ).

Special contrasts were computed, using the Bonferroni method, for all modality pairs. Significance levels for these contrasts are listed in Table 4. The visual ratings always differ significantly from both auditory and audiovisual ratings. In segments one and three, the difference between audiovisual and auditory ratings is non-significant or marginally significant. In segment two however, the audiovisual ratings differ significantly from both auditory and visual ratings. As seen in Figure 3, the visual channel seems to dominate first and the auditory channel thereafter.

### 3.1.3 Differenced ratings

The functional ANOVA analysis was repeated for differenced ratings, seen in the bottom panels of Figures 2, 3, and 4. The goal was to investigate, whether some of the findings in the first analysis would be due to an off-set rather than a profile difference. A further motivation for analysing differenced data is their reduced dependency on previous values. The results confirm the first analysis: modality had a significant main effect ( $p < 0.01$ ) but measurement did not ( $p \geq 0.19$ ). Nor was there a significant

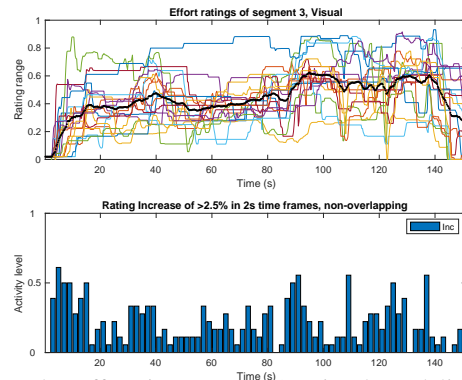


Figure 5: Effort in segment 3, visual modality. Top panel: individual and mean ratings (bold black line), bottom panel: activity levels for rating increases.

Original	AV-A	AV-V	A-V
Segment 1	* ( $p = 0.021$ )	**	***
Segment 2	***	***	**
Segment 3	-	***	***
Differenced	AV-A	AV-V	A-V
Segment 1	- ( $p = 0.059$ )	***	***
Segment 2	-	**	**
Segment 3	-	***	***

Table 4: Planned contrasts in functional analysis of variance for original and differenced data. Significance levels:  $p < 0.001$ \*\*\*,  $p < 0.01$ \*\*,  $p < 0.05$ \*,  $p > 0.05$ -.

interaction ( $p \geq 0.48$ ). Compared to the first analysis, the contrasts between audiovisual and auditory ratings in segments one and two are non-significant. These differences are therefore rather of the off-set type. This suggests that in terms of profile, the audiovisual ratings are closer to the auditory than the visual ratings.

In terms of distance, the (non-differenced) audiovisual ratings are not substantially closer to the auditory ratings or even to the channel whose ratings are higher. Rather, the dominant channel might be the one changing more radically. In this respect, segment two at 60-80 s is particularly interesting. There the ratings drop suddenly in all three conditions; however, audiovisual ratings seem to be captured by the steeper decline in the auditory channel.

## 3.2 Valence

Valence ratings were mostly uncoordinated. Only one of the nine factor combinations was significantly coordinated ( $C=3.38$  for auditory modality in segment three); for the rest, the C-Score was insignificant ( $C < 2$ ). While the auditory modality reached a mean C-Score not far from significant ( $C=1.84$ ), the visual and audiovisual modalities were clearly below it ( $C=0.68$  and  $C=0.83$ , respectively). An example of the ratings and activity levels in the visual modality is seen in Figure 6.

Figure 6 also shows great individual differences typical of the valence measurement. At the same time as some

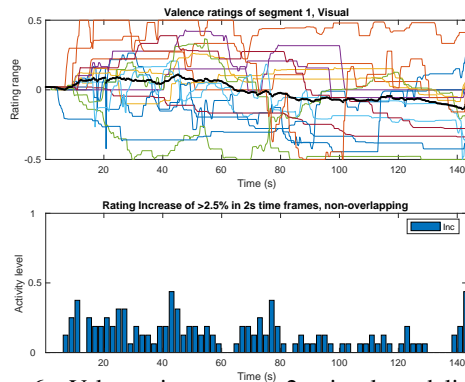


Figure 6: Valence in segment 2, visual modality. Top panel: individual and mean ratings (bold black line), bottom panel: activity levels for rating increases.

participants rated the passage as maximally positive, others gave maximally negative ratings. The resulting mean curve hardly deviates from zero. It is therefore impossible to make further conclusions based on the mean.

The reasons for the differences are of interest, however. As was seen, participants covered varying ranges. There was a positive association between participants' liking scores and mean valence levels taken across all their valence ratings (Pearson's  $r = 0.55$ ,  $p < 0.001$ ). Liking and familiarity were likewise positively associated ( $r = 0.51$ ,  $p < 0.001$ ). However, there was no significant association between mean valence levels and familiarity ( $r = 0.21$ ,  $p = 0.14$ ), nor between mean valence levels and being a musician ( $r = 0.09$ ,  $p = 0.51$ ).

## 4. DISCUSSION

### 4.1 Audiovisual perception

Arousal and effort were generally perceived with significant coordination between participants from auditory, visual, and audiovisual cues alike. Although both visual and audiovisual ratings were significantly coordinated in most of their factor combinations, only auditory ratings reached significant coordination in all cases. Participants also utilized a slightly wider range in the auditory ratings. Thus, auditory cues produced somewhat better alignment and higher variability in the rating curves than visual cues.

Significant bi-coordination was generally lacking between the unimodal auditory and visual conditions, and functional analysis of variance detected a significant difference between them in both original and differenced ratings. This outcome is in line with the study by Vines and colleagues, who observed significant differences between auditory and visual perception of musical tension [12]. They concluded that audiovisual ratings were dominated by the auditory modality. Here, such evidence is less conclusive. Even though the audiovisual rating profiles match better with the auditory ratings, the Activity Analysis confirms that both channels significantly drive audiovisual perception.

The theory of optimal sensory integration predicts that bimodal perception is dominated by the modality deliver-

ing more reliable information [25–27]. The auditory channel could be labelled generally more reliable, because it was perceived at a wider dynamic scale and thus contains more noticeable changes. One might argue that this difference was due to less realistic visual playback through a small screen. Literature on the effect of screen size on perceived affect is not exhaustive; iPhone-size screens are known to reduce immersion [28, 29]. It is indeed possible that visual perception might have been underestimated here in comparison to a hypothetical live concert situation. Live measurements, as a function of seating and lighting conditions in the hall, would be necessary to estimate the effect. Given that even with the current setup, mean visual ratings exceeded mean auditory ratings ca. 35% of the time, the effect of video presentation should be but small. Also various pilot experiments measuring effort, made using variable bigger screen sizes, delivered results similar to the current experiment. Moreover, one would expect the suppressing effect to be a negative off-set. The current results indicate, however, that the main reason why participants were more tuned to the auditory ratings were profile differences.

Analysis of the profiles in terms of differenced ratings suggests that audiovisual perception may have been captured by change. In this experiment, more change was generally perceived in the auditory channel, thus it transmitted more information than the visual channel. This seems only natural, given that the sound of musical instruments is supposed to respond to even very fine adjustments in playing gestures [30]. Moreover, because this study focused on the perception of basic affects, the material did not contain additional expressive gestures typical of traditional cello performance. If added, these might significantly increase the amount of movement, which could have an effect on perceived arousal along with other emotional cues. The current measurement therefore addresses a baseline of perceived audiovisual arousal, which could be generalized to both contemporary and classical cello repertoire containing mainly sound-producing movements. As a further step, the effect of expressive gestures on perceived arousal would be of interest.

As an immediate next step, visual attributes underlying arousal and effort perception is a research interest following this study. Intensity, and to some extent spectral centroid and spectral flatness, are known auditory cues for arousal [5, 31, 32]. Visual cues, movement size particularly, were found to convey loudness but not tempo changes in piano performance [21]. The first step is to investigate, how much of the variation in auditory and especially visual ratings is explained by intensity. In the visual domain, this should depend on the way loudness is controlled in an instrument. In bowed string instruments, loudness is increased through bow velocity, requiring larger and faster movements. The mapping between movement size and intensity is obvious also in the piano, but less so in wind instruments, the organ, and the harpsichord. Would the visual channel transmit even fewer bits of information in these cases, and lose importance? Or would the judgment require expertise on the instrument?

## 4.2 Arousal vs effort

Arousal and effort were bi-coordinated in all factor combinations, and no main effect of measurement was detected by functional analysis of variance. It is probable that participants judged both from the same cues, the main difference being the viewpoint. One of the goals of this study was to investigate, whether effort and arousal are perceived differently through auditory and visual channels. While exertion can be relatively well estimated from visual cues [33, 34], it was hypothesized that judging it from auditory cues might require more inference and be more difficult. Such differences were not observed. Non-musicians perceived on average nine percentage points more effort than musicians in the visual condition. However, the nature and significance of this potential effect cannot be estimated from the current data.

According to the FEELA hypothesis, the player's exertion is transmitted through intensity changes to perceived arousal [5, 7]. This implies that effort cues must also be arousal cues. Whether the contrary is true must depend on the circumstances. As long as the intensity changes are caused by the performer's exertion, effort and arousal perception should match well. This seems evident in the present case. Further experiments could be designed using music where intensity changes do not originate from effort changes, such as looming motion, or music with no intensity changes.

## 4.3 Valence

The valence measurement lacked significant coordination. This was not surprising, given the absence of culturally learned cues. There was a significant positive association between participants' individual mean valence level and their self-reported liking of the repertoire. Furthermore, valence ratings predicted the liking scores in two segments. Thus, subjects' attitude may have influenced their perception of one of the two basic affect dimensions. The influence of mood on perception has previously been observed using visual emotional stimuli [35]. Hence it is concluded that valence was not a reliable measure of emotional response to the repertoire in this study. However, detailed analysis of the valence ratings may reveal moments of high activity, even if the general coordination scores were low. Such moments may be of interest considering covariance with the other two measures, and will be analyzed at a later stage.

## 5. CONCLUSIONS

Perception of basic affect dimensions from auditory and visual cues was measured in cello performance. While the auditory and visual channels were perceived differently, ratings were generally significantly coordinated within each modality. Arousal and effort were perceived similarly, supporting the notion that in acoustic music performance, both are associated with loudness changes.

Auditory and visual ratings of arousal and effort were different but both influenced audiovisual perception. It is hypothesized that subjects' attention was caught by the more

dramatically changing modality. In the present case, more change was perceived in the auditory channel. This might be due to contents of the visual channel; however, an additional effect of transmission medium (screen size) cannot be eliminated. However, if the visual component is present in either live or recorded performance, both auditory and visual channels should ideally communicate a roughly equal amount of information in order not to suppress audiovisual perception. The present study concerns basic affect perception; in literature, visual cues have been shown to communicate higher level features, such as expressiveness or expertise, perhaps even more reliably than auditory cues.

Valence perception was uncoordinated between participants. Individual mean valence ratings were associated with participants' liking of contemporary music.

## Acknowledgments

Jan Schacher and Ellen Fallowfield are gratefully acknowledged for producing the musical material used in this experiment.

## 6. REFERENCES

- [1] M. Zentner and T. Eerola, "Self-report measures and models," in *Handbook of Music and Emotion: Theory, Research, Applications*, P. Juslin and J. Sloboda, Eds. Oxford University Press, 2010.
- [2] P. Juslin and R. Timmers, "Expression and communication of emotion in music performance," in *Handbook of Music and Emotion: Theory, Research, Applications*, P. Juslin and J. Sloboda, Eds. Oxford, UK: Oxford University Press, 2010.
- [3] I. Peretz, L. Gagnon, and B. Bouchard, "Music and emotion: perceptual determinants, immediacy, and isolation after brain damage," *Cognition*, vol. 68, no. 2, pp. 111–141, 1998.
- [4] K. N. Olsen, R. T. Dean, and C. J. Stevens, "A continuous measure of musical engagement contributes to prediction of perceived arousal and valence," *Psychomusicology Music. Mind, Brain*, vol. 24, no. 2, pp. 147–156, 2014.
- [5] R. T. Dean and F. Bailes, "Time series analysis as a method to examine acoustical influences on real-time perception of music," *Empir. Musicol. Rev.*, no. 4, pp. 152–175.
- [6] R. Dean and F. Bailes, "A Rise-Fall Temporal Asymmetry of Intensity on Composed and Improvised Electroacoustic Music," *Organ. Sound*, vol. 15, no. 2, pp. 147–158, 2010.
- [7] K. N. Olsen and R. T. Dean, "Does Perceived Exertion Influence Perceived Affect in Response to Music? Investigating the FEELA Hypothesis," *Psychomusicology Music. Mind, Brain*, vol. 26, no. 3, pp. 257–269, 2016.

- [8] A. Camurri, G. De Poli, A. Friberg, M. Leman, and G. Volpe, "The MEGA Project: Analysis and Synthesis of Multisensory Expressive Gesture in Performing Art Applications," *J. New Music Res.*, vol. 34, no. 1, pp. 5–21, 2005.
- [9] C.-J. Tsay, "Sight over sound in the judgment of music performance," *Proc. Natl. Acad. Sci. USA*, vol. 110, no. 36, pp. 14 580–14 585, 2013.
- [10] M. Rodger, C. Craig, and S. O'Modhrain, "Expertise is perceived from both sound and body movement in musical performance," *Hum. Mov. Sci.*, vol. 31, no. 5, 2012.
- [11] H. F. Mitchell and R. a. R. MacDonald, "Listeners as spectators? Audio-visual integration improves music performer identification," *Psychol. Music*, vol. 42, pp. 112–127, 2012.
- [12] B. W. Vines, C. L. Krumhansl, M. M. Wanderley, and D. J. Levitin, "Cross-modal interactions in the perception of musical performance," *Cognition*, vol. 101, no. 1, pp. 80–113, 2006.
- [13] W. Thompson, P. Graham, and F. Russo, "Seeing music performance: Visual influences on perception and experience," *Semiotica*, vol. 156, pp. 177–202, 2005.
- [14] M. Broughton and C. Stevens, "Music, movement and marimba: an investigation of the role of movement and gesture in communicating musical expression to an audience," *Psychol. Music*, vol. 37, no. 2, pp. 137–153, 2009.
- [15] C. J. Stevens, E. Schubert, R. H. Morris, M. Frear, J. Chen, S. Healey, C. Schoknecht, and S. Hansen, "Cognition and the temporal arts: Investigating audience response to dance using PDAs that record continuous data during live performance," *Int. J. Hum. Comput. Stud.*, vol. 67, no. 9, pp. 800–813, 2009.
- [16] M. R. Thompson, M. M. Wanderley, and G. Luck, "Deadpan and immobile performance intentions share movement features but not expressive parameters," in *Proc. ICMPC and the 8th Conf. Eur. Soc. for Cogn. Sci. of Mus.*, Thessaloniki, 2012, pp. 987–988.
- [17] S. Dahl and A. Friberg, "Visual perception of expressiveness in musicians' body movements," *Music Perc.*, vol. 24, pp. 433–454, 2007.
- [18] W. Thompson, F. Russo, and L. Quinto, "Audio-visual integration of emotional cues in song," *Cogn. Emot.*, vol. 22, no. 8, pp. 1457–1470, 2008.
- [19] A. Friberg, "A fuzzy analyzer of emotional expression in music performance and body motion," in *Proc. Music and Music Science*, Stockholm, 2004, pp. 1–13.
- [20] J. Vuoskoski, M. Thompson, E. Clarke, and C. Spence, "Crossmodal interactions in the perception of expressivity in musical performance," *Atten. Percept. Psychophys.*, vol. 76, no. 2, pp. 591–604, 2014.
- [21] J. Vuoskoski, M. Thompson, C. Spence, and E. Clarke, "Interaction of sight and sound in the perception and experience of musical performance," *Music Percept.*, vol. 33, no. 4, 2016.
- [22] F. Upham and S. McAdams, "Activity Analysis and Coordination in Continuous Responses to Music," *Music Percept. an Interdiscip. J.*, vol. 35, no. 3, pp. 253–294, 2018.
- [23] M. Febrero-Bande and M. O. de la Fuente, "Statistical computing in Functional Data Analysis: the R package fda.usc," *J. Stat. Softw.*, vol. 51, no. 4, 2012.
- [24] J. A. Cuesta-Albertos and M. Febrero-Bande, "A simple multiway ANOVA for functional data," *Test*, vol. 19, pp. 537–557, 2010.
- [25] M. O. Ernst and M. Banks, "Humans integrate visual and haptic information in a statistically optimal fashion," *Nature*, vol. 415, 2002.
- [26] D. Alais and D. Burr, "The Ventriloquist Effect Results from Near-Optimal Bimodal Integration," *Curr. Biol.*, vol. 14, pp. 257–262, 2004.
- [27] M. Ernst, "A Bayesian view on multimodal cue integration," in *Hum. Body Percept. from Inside Out*, G. Knoblich, I. Thornton, M. Grosjean, and M. Shiffrar, Eds. New York: Oxford University Press, 2006, pp. 105–131.
- [28] J. Hou, Y. Nam, W. Peng, and K. M. Lee, "Effects of screen size, viewing angle, and players' immersion tendencies on game experience," *Comput. Human Behav.*, vol. 28, pp. 617–623, 2012.
- [29] J. Rigby, D. Brumby, A. Cox, and S. Gould, "Watching movies on netflix," in *Proc. 18th Int. Conf. Human-Comp. Interact. with Mob. Devices Serv. Adjunct. - MobileHCI '16*. New York: ACM Press, 2016, pp. 714–721.
- [30] S. O'Modhrain and R. B. Gillespie, *Once More, with Feeling: Revisiting the Role of Touch in Performer-Instrument Interaction*. Cham: Springer International Publishing, 2018, pp. 11–27.
- [31] R. Dean and F. Bailes, "Modelling Perception of Structure and Affect in Music: Spectral Centroid and Wishart's Red Bird," *Empir. Musicol. Rev.*, vol. 6, no. 2, 2011.
- [32] R. T. Dean, F. Bailes, and E. Schubert, "Acoustic intensity causes perceived changes in arousal levels in music: an experimental investigation," *PLoS One*, vol. 6, no. 1, pp. 1–8, 2011.
- [33] J. Shim, L. G. Carlton, and J. Kim, "Estimation of Lifted Weight and Produced Effort through Perception of Point-Light Display," *Perception*, vol. 33, no. 3, pp. 277–291, 2004.



- [34] M. Auvray, T. Hoellinger, S. Hanneton, and A. Roby-Brami, "Perceptual weight judgments when viewing one's own and others' movements under minimalist conditions of visual presentation," *Perception*, vol. 40, pp. 1081–1103, 2011.
- [35] J. Jolij and M. Meurs, "Music alters visual perception," *PLoS One*, vol. 6, no. 4, 2011.

# Dancing Dots - Investigating the Link between Dancer and Musician in Swedish Folk Dance.

**Olof Misgeld**

Royal College of Music (KMH)  
and  
KTH Royal Institute of Technology  
misgeld@kth.se

**Andre Holzapfel**

KTH Royal Institute of Technology  
holzap@kth.se

**Sven Ahlbäck**

Royal College of Music (KMH)  
sven.ahlback@kmh.se

## ABSTRACT

The link between musicians and dancers is generally described as strong in many traditional musics and this holds also for Scandinavian Folk Music - *spelmansmusik*. Understanding the interaction of music and dance has potential for developing theories of performance strategies in artistic practice and for developing interactive systems. In this paper we investigate this link by having Swedish folk musicians perform to animations generated from motion capture recordings of dancers. The different stimuli focus on motions of selected body parts as moving white dots on a computer screen with the aim to understand how different movements can provide reliable cues for musicians. Sound recordings of fiddlers playing to the "dancing dot" were analyzed using automatic alignment to the original music performance related to the dance recordings. Interviews were conducted with musicians and comments were collected in order to shed light on strategies when playing for dancing. Results illustrate a reliable alignment to renderings showing full skeletons of dancers, and an advantage of focused displays of movements in the upper back of the dancer.

## 1. INTRODUCTION

Research on interactions between participants in a music performance can reveal basic strategies and help to develop theories of how performances are shaped in specific performance contexts. In specific here, the focus lies on the interaction between a musician and a dancer and how musicians shape performance when interpreting dance movements. In terms of computational applications, an understanding of how these interactions work is an important basis for the implementation of interactive systems, as for instance the automatic alignment of a music performance recording to the movements of a dancer, or – thought the other way around – the generation of a virtual dancer to the real-time performance of a musician. In terms of artistic research, revealing tacit strategies for how body movement can be interpreted musically has great potential for exploring new

artistic concepts that can be used in interaction between music and dancers.

In this paper, we focus on the specific performance context of Swedish Folk Music, *spelmansmusik* and the music and dance form *polska*. In this musical form, as in many traditional musics, music and dance are usually performed together, which creates strong links between involved performers [1]. Considering the dance and music as complementary elements has been common to the study and understanding of this performance practice [2–5]. The shaping of musical beat in music and in dance has been pointed to as central for this interaction and for how different metrical types, styles and forms are developed. [2, 6, 7].

Research into musicians' movement in classical Euro-genetic [8] music documented cues and gestures in ensemble and solo playing [9–11]. Studies of Scandinavian folk music have begun to explore how embodied meter is expressed in motion patterns of dancers and musicians [5] and in relation to musical beats [12]. It remains an open question what role the visual contact between musicians and dancers plays for this interaction, for instance if a musician's focus on certain body parts of a dancer may facilitate a more stable interaction, or if generally the perspective on the whole body is needed. This can also be related to ways of describing dance heritage documentation and in folk dance didactic practices [13].

In this paper, we explore if the reduction of a dancers' movement obtained through infrared Motion Capture technology (MoCap) still facilitates the performance by a musician when viewing this reduced movement. Although this setup does not include the full interaction between player and dancer the experiment will help the understanding of which aspects of body movements provide reliable cues for players, and how reliable these cues are. To this end, performance recordings were used that had been conducted [12] with an Optitrack MoCap system, with three fiddle-playing musicians and two dancers forming six pairs of musician and dancer, playing two pieces of a local polska tradition. For the experiments in this paper, five different visualizations of the dancers' body movements were generated that visualize movements of various body parts. The three musicians taking part in the initial performances were asked to play the related pieces only to the generated visual cues, without sound cues accompanying the display. The obtained outcomes provide first results of both qualitative and quantitative nature of how the musicians are able

Copyright: © 2019 Olof Misgeld et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

to synchronize to the various cues. The results indicate that when viewing a full body rendering of the dancer, the players were able to follow the dance almost perfectly. In cases of more radical reductions of the dance movements - using only one or two markers from the dancers - the task became harder, resulting in increased deviations. However, cases where players were aligned to their original performance demonstrate that the reduced renderings still contain various degrees of information for the player.

The remainder of this paper is structured as follows: Section 2 provides a concise overview of the related literature, including studies of interactions in music performance, and in particular of interactions between dancers and musicians. Section 3 describes the construction of the stimuli, the experimental setup, and the analysis methods. The results are provided in Section 4 and discussed with suggestions for further research in Section 5.

## 2. BACKGROUND

Various studies have documented the human ability to recognize and relate to human motions from highly reduced representations. Research with point-lights by Johansson [14] and Cutting et al [15] showed perception of human walking and recognition of individual walking styles from a small set of moving dots. Point-lights were also used by Pollick et al [16] to study the perception of gender and affect from isolated hand movements and by Petrini et al [17] to study the effect of musical expertise on the sensitivity to asynchrony of drummers.

Synchronization in music ensembles was studied, for instance, by Hofmann et al [18], who analyzed how the timing of an individual performer influences ensemble synchronization in Jazz trio performances. In different musical context, Bishop et al [10] studied the synchronization of classical duo players (violin and piano) using only visual or only audio signals. MoCap was used by Keller et al [11] when examining the effect of anticipatory auditory imagery - described as the skill of planning and predicting actions during a musical performance - on the temporal coordination of body movement and sound in classical piano duos.

Toivianinen et al [19] used MoCap to study dance movements in relation to musical structure - how different metrical levels were manifested in different parts of human bodies moving to music. Naveda et al [20] suggested a model of topological gesture analysis that related dancers' motions to musical and metrical patterns in a spatiotemporal representation. Scandinavian Folk dance styles have been characterized by the vertical motion patterns of the center of gravity (libration curve/sviktkurva) by Blom [2] which influenced research and didactic practise of Scandinavian folk dance. This synchronization between dance and music has been further explored using MoCap in settings with interacting dancers and musicians [5, 12]. To the best of our knowledge, synchronization of instrument playing to reduced renderings of dance movement have not been studied before.

In the context of sound and music computing, systems for tracking periodic movements have been developed and

implemented for sports and wellbeing [21, 22], and motion sonification systems have been developed for applications within circus, dance and opera [23]. A categorization of musical gestures into sound-producing (by musicians) and sound-perceiving (e.g. by dancers) was suggested by Jensenius et al. [24]. However, such a division between gestures can be blurred in situations where dancers may influence the playing of a musical instrument by means of their movement. Our study will therefore contribute to investigations of the rhythmic gestures that emerge from the periodic movements of dancers.

## 3. METHOD

### 3.1 Material

In an earlier study [12], three musicians playing for two dancers were recorded in an Optitrack MoCap System. The recordings included settings with each musician playing the same two pieces to each of the dancers dancing alone or together in a couple. 41 markers were placed on each dancers to facilitate a close rendering of the dancers full body movements. Musical beats were manually annotated from the audio of the recordings to allow comparisons with the movement data.

For the present study recordings from this material were selected in settings where each of the two dancers were dancing alone to each of the three players. Each of these settings had recordings with the same two music pieces resulting in a total number of 12 recordings (2 pieces  $\times$  2 dancers  $\times$  3 players). The pieces used in the study were two *polska* tunes referred to as *Lorikspolskan* and *Polska efter Pellar Anna* related to the influential fiddler Gössa Anders Andersson (1878-1962) from Orsa in Dalarna, Sweden [25]. These tunes are quite complicated in terms of rhythmic and metrical structure, with varying beat duration, obscured beat onsets and varying rhythmic figures [7]. Five different types of stimuli were generated from each recording using the MoCapToolBox [26]. These stimuli consisted of videos showing five different renderings of the MoCap data with markers moving as white dots on a black background.

The five renderings were chosen to reflect two strategies when watching the dance: focusing on the feet and steps, or focusing on the general movement of the center of the body. The first type of stimuli videos (labeled BT) included one single marker placed on the spine below the neck, and included both y-axis (vertical), and a projection of x- and z-axis on a plane, transforming the circular movement of the dancer into a horizontal movement on the screen. The second video showed the same marker moving only vertically, as this rendering only included movement data from the y-axis (labeled BTY). The third and fourth videos showed two markers placed on the dancers feet, close to the joint of the little toe, in similar renderings to the first and second stimuli. These stimuli will be labeled RLT, for those animations including y-axis and the projection of x- and z-axis of left and right foot, and RLTY, for those animations including y-axis only. The fifth video showed all 41 mark-

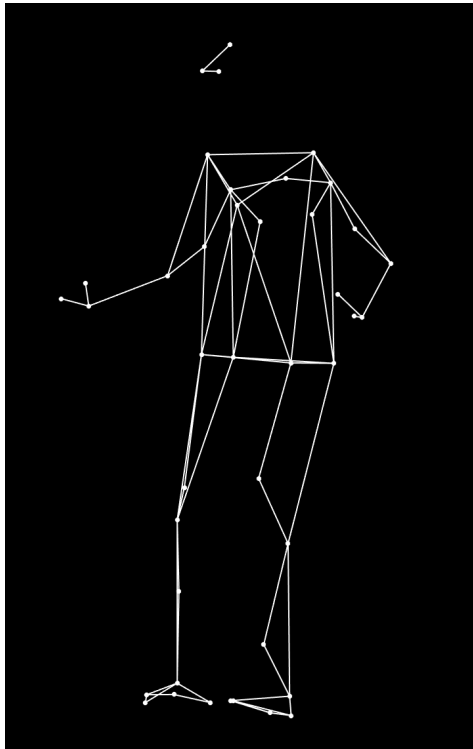


Figure 1. The dancer projected in the FULL stimuli.

ers of the dancer connected with thin white lines in order to form a skeleton of the dancers body was labeled FULL and is displayed in Figure 1. This full marker setup was included to investigate the impact of the reduction present in the transformation of the whole body to a set of limited interconnected points. The stimuli were generated using a frame rate of 30 frames per second.<sup>1</sup>

To facilitate comparison of the audio recordings of the players performing to the animation (referred to as *secondary* recordings in the remainder of the text) to the original audio (*i.e.* performing with the dancing as recorded in the context of [12]), a reference was provided to inform players where to start playing. This was achieved by including a small part of the sound from the original recording as a cue in the beginning of the videos. The included part consisted of two measures of the original recorded tune. Initial clap sounds were added to the animation videos to allow for precise synchronization of the performance beginnings, and the investigation of the alignment of the original and secondary recordings.

### 3.2 Participants

Participating as performers in the present study were musicians Sven Ahlbäck, Ellika Frisell and Olof Misgeld. All three participating musicians have 25-45 years experience as performers within this style, and 15-25 years as teachers of Folk Music within higher performance education at the Royal College of Music in Stockholm (KMH). They are identical with the players who recorded the original per-

formances [12], and they were asked to perform to the five types of animations emerging from the four performances of themselves playing two tunes to two dancers in the original recordings (2 pieces  $\times$  2 dancers  $\times$  5 animation types = 20 stimuli per player). The dancers in the original recordings (Ami Dregelid and Andreas Berchthold) have a high level of experience as performers and dance pedagogues.

### 3.3 Experimental setup

The experimental setup had each musician performing seated in front of a computer screen showing the animations. The performances were recorded using two microphones, one directed towards the violin, and one towards the feet in order to obtain clear sound recordings of the foot tapping. All session were recorded on video to capture the players' comments between performing to the stimuli. Before each stimulus, the player was provided with the information which of the two polskas was used in the video they were watching. The players were instructed to play the same piece by trying to synchronize their playing using the information from the stimuli. Both tunes consisted of two repeated parts forming a complete round (AABB), and each round was repeated two or three times. The players were informed that they will be playing to stimulus that emerged from a dance performance they were originally involved in as musicians, but they were not told which of the dancers was dancing in each stimulus. In the experiment all five animation types related to one recorded piece were presented before moving on to the next piece. This was done in the order of starting from the four more reduced renderings and finishing each take with the full skeleton. No written music was used at any point as pieces were all played from memory.

### 3.4 Analysis

For the analysis, the beginning of each secondary recording was manually aligned to the beginning of the original recording using the recorded clap sounds. The recordings were then compared by listening simultaneously to the two files, playing one file in each ear while making annotations on how recordings diverged, noting stable or diverging sections. Where a diversion was noted sample measurements in seconds were estimated from corresponding beats in the two recordings.

An automatic alignment of the secondary recordings to the original recordings was created using an audio matching algorithm [27]. The outputs of the automatic alignment were compared to the manual annotations, with the automatic alignment being consistent with manual annotations in all cases. Based on the reliability of the automatic alignment, alignment curves were computed for all pairs of original and secondary sound recordings. The curves specify the temporal shift in seconds of each point in the original recording compared to the secondary recording (see Figure 2). The mean beat duration obtained from beat annotations of the original recordings enable us to relate this temporal shift in seconds to beats. Using the alignment curves, the recordings were divided into segments belonging to one of three categories:

<sup>1</sup> Examples of stimuli videos are provided in <https://bit.ly/2E8jxte>



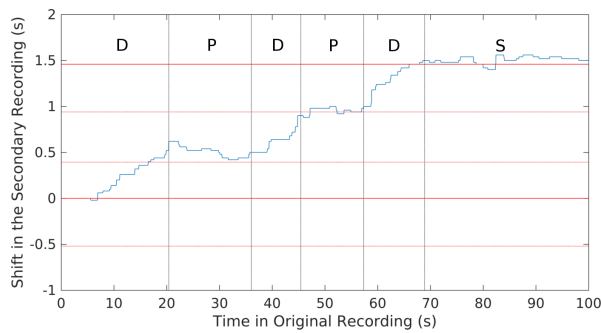


Figure 2. Example of an alignment curve (BT stimulus), showing a stable alignment after 68s (S), and changes between drift (D) and alignment to other metrical levels (P) before. The bold horizontal lines mark measure positions, whereas dotted horizontal lines mark the second and third beats of the three-beat cycle, as obtained from the manual beat annotations.

1. **STABLE:** segments where the alignment curve stays within the area of  $\pm$  one beat to the 0-line, without moving monotonously to a neighboring beat annotation and crossing it. Also included into this category were segments where the alignments curves shifted by a multiple of a measure, which occur after a player had been drifting in tempo in a previous segment or if the player changed the form of the piece. STABLE segments were taken as indication that the player relates in a stable way to the dance animation. For example, the segment between 68s and 100s in Figure 2 was marked as STABLE, since it is shifted by one complete measure.
2. **DRIFT:** segments where the alignment curve moves monotonously across beat annotations. These phases are seen as an indication that the player was not able to synchronize her/his playing to the stimulus in a stable way, which resulted in slowing or speeding up.
3. **PHASE-SHIFTED:** segments where the alignment curve stays within the area of  $\pm$  one beat to a non-measure line (dotted horizontal lines in Figure 2), without monotonous drift to a neighboring beat annotation and crossing it. This case is related to shifting one or two beats to the original. This was interpreted as the player adhering to a different interpretation of the three-beat cycle of the *polska*, something unlikely to occur in real performance settings.

The sessions were video recorded, and semi-structured interviews were conducted with the performers between the experiment tasks. Statements of the players of the qualitative experiences will be used in addition to the quantitative measures obtained from annotating the alignment curves. This helps to identify the strategies that were applied by the three performers in following the visualizations.

Stimulus type	RLT	RLTY	BT	BTY	FULL
Stable (%)	49.7	46.9	72.3	73.1	98.4
Drift (%)	34.4	36.7	19.2	23.5	1.6

Table 1. Share of the segments annotated as either stable or drifting, as percentages of the whole duration of recorded secondary performances (about 1h22min, equally divided among the five stimulus types).

## 4. RESULTS

### 4.1 Automatic alignments

Following the annotations of STABLE segments, and to DRIFT and PHASE-SHIFT segments as described in Section 3.4, the percentage of STABLE and DRIFT phases occurring in all secondary recordings was calculated (Table 1). High shares of STABLE phases indicate that performers synchronized well to a particular stimulus type, whereas high shares of DRIFT phases indicate that the performers were not able to extract reliable tempo-related information from the stimulus type.

Stable-aligned segments cover almost the complete duration (98.4%) of recorded secondary performances for the FULL setup. This demonstrates that when seeing the full skeleton, the players were almost perfectly capable to align their performance with the stimulus. This extends previous results of movement interpretation [14, 15] to the task of performance synchronization, and provides a proof of concept for the validity of the motion capture data. Performers were able to synchronize to a lesser degree to the markers on the neck (BT, BTY), and even less to the visualizations obtained from the feet (RLT, RLTY). Overall, however, performers could synchronize in a stable way in at least 46.9% of the recordings (RLTY), which indicates that all visualizations provide important information on the dancers' movement. The quality of the alignment in the stable phases, computed as the average area under the alignment curve, is quite high, with the performers synchronizing with an average deviation of about 80ms, independent from the visualization type.

For the DRIFT phases, an opposite trend emerges from the second line of Table 1. Full body skeleton visualizations lead to almost no drift phases (1.6%), and RLT/RLTY have the highest amount of phases in which players are not able to synchronize to the tempo of the initial performance. Even though statistical significance could not be reached in the given amount of performances, the visualization of the marker on the neck (BT, BTY) enables for a more stable alignment than the visualization of the feet. This finding corroborates the approach of using the center of gravity for analysis of Nordic folk dance [2, 5].

### 4.2 Comments on playing with dots

After completing each set of five stimuli, musicians were asked to compare the task of playing to the different stimuli videos. Players generally commented that watching and having to follow the dance this closely was unusual as in normal situations they would rely on interacting with the

dancer, as one player said similar to being in a conversation with the dancer. The FULL stimuli were commented on as easiest and most similar to a real situation, and players were also able to quickly identify which dancer were dancing when watching the full-body skeleton.

After the FULL stimuli, the BTY stimuli were generally commented as second-easiest. This was explained by being able to focus on only one dot moving in the vertical direction. Also mentioned was the accelerating movements to and from the beats as the dot was in constant motion. These accelerations were also included in the BT stimuli, but then players stated that it was easier to lose track as the dot was moving horizontally as well.

The RLT and RLTY stimuli were commented on as more difficult, and players reported being confused and losing track. Players expressed a drifting out of phase when dancers changed steps between the walking and turning sections of the polska dance. RLT and RLTY were also commented as more static, showing more interrupted movements, and not consistently relating to tempo, phase and period.

These qualitative outcomes are consistent with the quantitative findings summarized in Table 1. In addition, players commented that it generally became easier to play to the stimuli as they got more used to them and learned to interpret them. One player commented he/she thought all renderings had the potential to work after a certain amount of practice.

#### 4.3 Strategies when playing for dancing

Strategies when playing for dancing can include what aspect of the dancer's motion the player is focusing on when watching the dance. Players commented on this in relation to the FULL video saying they were considering the interaction between different body parts taking into account changes in inclination, acceleration and balance of the body. One player commented on this as how the dancer is dealing with gravitation. Hand movements were also commented on as having a relation to the played tune: as if the dancers were playing the melody.

When trying to recognize the two dancers from more reduced renderings, the players commented on individual differences in dancing styles. They also related such differences to dancers typically dancing on either the left or the right side when dancing in pairs. Strategies included identifying a certain beat in the three-beat polska cycle. This was recurrently commented on in relation to playing to reduced renderings, i. e. finding the second beat in a lifting motion or the first beat in a more concise marked motion. Focusing on a certain beat in the animation was commented on as affecting the performance, inspiring players to emphasize a certain beat or choose a certain interpretation of asymmetric beat patterns [12].

Another strategy concerned the foot-tapping. A player commented that he/she refrained from tapping strongly in order to be more receptive when watching the stimuli. Normally he/she would think of the tapping as a kind of motor for the music.

Players commented that when playing for groups in more diverse real-life settings they take into account that steps

are not always synchronized with all beats in the music. Players stated that they would often benefit from watching the feet/steps, however not in isolation but with attention to the whole dancer and the transfer of weight when moving. Subsequently the renderings of RLT and RLTY do not compare entirely to a situation when watching the feet of a real dancer.

## 5. CONCLUSIONS

The results of this experiments demonstrate that musicians are able to interpret and synchronize to dancers movements also from reduced renderings. Both qualitative and quantitative results emphasize – on the one hand – the importance of having information from the whole body, but – on the other hand – demonstrate the large amount of information still present in a very radical reduction, especially for the BLT and BLTY stimuli. Previous studies of movement interpretations from point-lights did to our knowledge not include these kind of settings.

The question remains to what extent this is an effect of expertise within the specific performance context, which could be tested by extending experiments to performers with less experience in accompanying Swedish folk dance. The experiments provided a challenge since players rarely play to pre-recorded dancing without the possibility to interact. Visualizations with only one or two markers added to the challenge as musicians had to interpret a very limited information from a dancer.

The study suggests that music-dance interaction is far from trivial to the performers in this context, and that the required embodied knowledge should be further explored in combined scientific and artistic research settings. Further insights can pave the way to facilitate applications that allow dancers to interact with pre-recorded music, or musicians with a virtual dancer. The goal of such applications is seen in the extension of current performance possibilities, not in the replacement of the interaction between musicians and dancers.

## Acknowledgments

We thank the musician Ellika Frisell, and the dancers Andreas Berchthold and Ami Dregelid for making this study possible through their participation in studies and discussions of the results.

## 6. REFERENCES

- [1] T. Turino, *Music as social life: The politics of participation*. University of Chicago Press, 2008.
- [2] J.-P. Blom, "Rytme og frasering - forholdet til dansen," in *Fanitullen. Innføring i norsk og samisk folkemusikk*, B. Aksdal and S. Nyhus, Eds. Universitetsforlaget, 1993, pp. 161–184.
- [3] S. Ahlbäck, "Har man olika relation till musik i folkdans och konstdans?" in *Dansens dimensioner*, E. Bakka, Ed. Trondheim: Rådet för folkemusikk og folkedans, 1993, pp. 15–23.

- [4] M. Johansson, "Empirical research on asymmetrical rhythms in scandinavian folk music: A critical review," *Studia Musicologica Norvegica*, vol. 43, no. 01, pp. 58–89, 2017.
- [5] M. R. Haugen, "MusicDance Investigating Rhythm Structures in Brazilian Samba and Norwegian Telespringar Performance," Doctoral thesis, Universitetet i Oslo, 2016.
- [6] K. Eriksson and M. Nilsson, "Ethnomusichoreology? Ethnochoreomusic?" in *Crossing Over, Fiddle and Dance Studies from around the North Atlantic 3*, I. R. Guign and A. Kearney, Eds. Aberdeen: The Elphinstone Institute, 2010.
- [7] S. Ahlbäck, "About Asymmetrical Beat in the Polska," in *The Polish Dance in Scandinavia and Poland*, M. Ramsten, Ed. Stockholm: Svenskt visarkiv, 2003, pp. 165–80.
- [8] R. F. Reigle, "A brief history of kurdish music recordings in turkey," *Hellenic Journal of Music, Education and Culture*, vol. 4, no. 1, 2014.
- [9] E. Coorevits and D. Moelants, "Decomposing a Composition: On the Multi-layered Analysis of Expressive Music Performance," vol. 9617, no. June, 2016.
- [10] L. Bishop and W. Goebel, "When they listen and when they watch: Pianists' use of nonverbal audio and visual cues during duet performance," *Musicae Scientiae*, vol. 19, no. 1, pp. 84–110, 2015.
- [11] P. E. Keller and M. Appel, "Individual Differences, Auditory Imagery, and the Coordination of Body Movements and Sounds in Musical Ensembles," *Music Perception: An Interdisciplinary Journal*, vol. 28, no. 1, pp. 27–46, 2010.
- [12] O. Misgeld and A. Holzapfel, "Towards the study of embodied meter in Swedish folk dance," in *Folk Music Analysis workshop*, no. Mid, 2018.
- [13] L. Helmersson, *Eldsjärlarna och dansarvet : om forskning och arbetet med att levandegöra äldre dansformer*. Rättvik: Folkmusikens hus, 2012.
- [14] G. Johansson, "Visual perception of biological motion and a model for its analysis," *Perception & Psychophysics*, vol. 14, no. 2, pp. 201–211, 1973.
- [15] J. E. Cutting and L. T. Kozlowski, "Recognizing friends by their walk: Gait perception without familiarity cues," *Bulletin of the Psychonomic Society*, vol. 9, no. 5, pp. 353–356, 1977.
- [16] F. E. Pollick, V. Lestou, J. Ryu, and S. B. Cho, "Estimating the efficiency of recognizing gender and affect from biological motion," *Vision Research*, vol. 42, no. 20, pp. 2345–2355, 2002.
- [17] K. Petrini, S. Dahl, D. Rocchesso, C. H. Waadeland, F. Avanzini, A. Puce, and F. E. Pollick, "Multisensory integration of drumming actions: Musical expertise affects perceived audiovisual asynchrony," *Experimental Brain Research*, vol. 198, no. 2-3, pp. 339–352, 2009.
- [18] A. Hofmann, B. C. Wesolowski, and W. Goebel, "The tight-interlocked rhythm section: Production and perception of synchronisation in jazz trio performance," *Journal of New Music Research*, vol. 46, no. 4, pp. 329–341, 2017.
- [19] P. Toiviainen, G. Luck, and M. R. Thompson, "Embodied meter: hierarchical eigenmodes in music-induced movement," *Music Perception: An Interdisciplinary Journal*, vol. 28, no. 1, pp. 59–70, 2010.
- [20] L. Naveda and M. Leman, "The Spatiotemporal Representation of Dance and Music Gestures using Topological Gesture Analysis (TGA)," *Music Perception*, vol. 28, no. 1, pp. 93–111, 2010.
- [21] J. A. Hockman, M. M. Wanderley, and I. Fujinaga, "Real-time Phase Vocoder Manipulation by Runner's Pace," *The 9th International Conference on New Interfaces for Musical Expression*, pp. 90–93, 2009.
- [22] B. Moens, C. Muller, L. van Noorden, M. Franěk, B. Celie, J. Boone, J. Bourgois, and M. Leman, "Encouraging spontaneous synchronisation with d-jogger, an adaptive music player that aligns movement and music," *PloS one*, vol. 9, no. 12, p. e114234, 2014.
- [23] L. Elblaus, M. Goia, M.-A. Robitaille, and R. Bresin, "Modes of sonic interaction in circus: Three proofs of concept," *Proceedings - 40th International Computer Music Conference, ICMC 2014 and 11th Sound and Music Computing Conference, SMC 2014 - Music Technology Meets Philosophy: From Digital Echos to Virtual Ethos*, no. September, pp. 1698–1706, 2014.
- [24] A. R. Jensenius and M. Wanderley, "Musical gestures : concepts and methods in research," no. January, 2009.
- [25] Musica Svecia, *Folk music in Sweden: Folk tunes from Orsa and Älvdalen*. Stockholm: Caprice Records, 1995.
- [26] B. Burger and P. Toiviainen, "MoCap Toolbox - A Matlab toolbox for computational analysis of movement data," *Proceedings of the Sound and Music Computing Conference 2013, SMC 2013*, pp. 172–178, 2013.
- [27] C. Cannam, M. O. Jewell, C. Rhodes, M. Sandler, and M. d'Inverno, "Linked data and you: Bringing music research software into the semantic web," *Journal of New Music Research*, vol. 39, no. 4, pp. 313–325, 2010.

# Conditioning a Recurrent Neural Network to synthesize musical instrument transients

**Lonce Wyse**

Communication and New Media  
National University of Singapore  
lonce.wyse@nus.edu.sg

**Muhammad Huzaifah**

NUS Graduate School for Integrative  
Sciences and Engineering  
National University of Singapore  
e0029863@u.nus.edu

## ABSTRACT

A Recurrent Neural Network (RNN) is trained to predict sound samples based on audio input augmented by control parameter information for pitch, volume, and instrument identification. During the generative phase following training, audio input is taken from the output of the previous time step, and the parameters are externally controlled allowing the network to be played as a musical instrument. Building on an architecture developed in previous work, we focus on the learning and synthesis of transients – the temporal response of the network during the short time (tens of milliseconds) following the onset and offset of a control signal. We find that the network learns the particular transient characteristics of two different synthetic instruments, and furthermore shows some ability to interpolate between the characteristics of the instruments used in training in response to novel parameter settings. We also study the behavior of the units in hidden layers of the RNN using various visualization techniques and find a variety of volume-specific response characteristics.

## 1. INTRODUCTION

When musical wind instrument sounds are initiated by blowing air through or across a mouthpiece, the time it takes for the system to reach a stable resonant state is referred to as an “attack” transient. When energy ceases to be put in to the system, the time it takes for the instrument to return to rest is a “decay” transient. The attack is typically complex with different frequency components reaching their steady states via different amplitude trajectories (“envelopes”). With physical instruments, the attack characteristics vary significantly across the way the instrument is articulated with tongue and breath, and across different instruments. The attack characteristics are an important perceptual indicator used by listeners to identify playing style and the instrument being played (Grey [1]).

Transients are interesting from a modeling perspective in part because of their non-instantaneous response to the articulation parameters. There is no simple mapping from breath pressure to the sound signal the instrument radiates, but there is rather a state-dependent temporal evolution of the sound that follows sudden changes in the parameters.

In our previous work, (Wyse [2]), we developed a recurrent neural network (RNN) for modeling musical in-

strument sound generation. RNNs were chosen because they are structured and often used to model sequences such as digital sound samples. The training was conditioned on pitch, volume, and instrument ID in addition to the audio stream so that during generation following training, the parameters could be used to control synthesis. However, in this previous work, only steady-state tones were used during training, so none of the specific instrument transient characteristics were learned.

In this paper, we train the network on two different synthetic instruments that, in addition to having different harmonic structures, also have distinct attack and decay times that follow sudden changes in the control parameter that we use as a proxy for breath pressure, and that we refer to herein as “volume”. We also explore the activations of hidden units in response to parameter changes during generation and find interesting patterns such as volume-specific response characteristics.

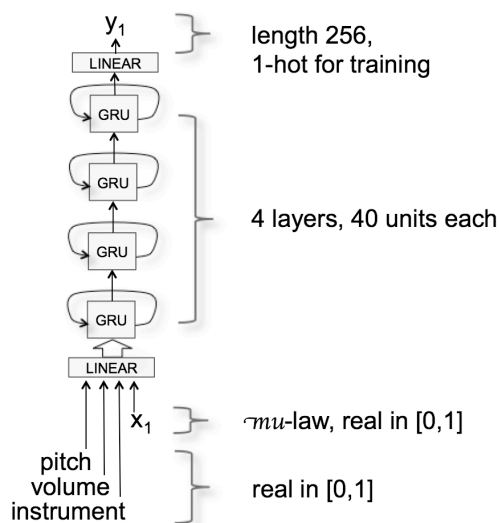
## 2. MODELING

### 2.1 Architecture

The architecture of the model is the same as that used in Wyse [2] and is summarized here (Figure 1). It is a stacked RNN composed of 4 layers of Gated Recurrent Units (GRU) (Cho et al. [3]) sandwiched between dense linear layers after the input and before the output.

The input is a vector of 4 components at each time step (sample rate=16000 Hz) representing the audio sample, the pitch, volume, and instrument ID. Audio is mu-law encoded with 256 values and normalized to [0,1], pitch consists of the 13 chromatic notes spanning the octave from E4 (fundamental frequency≈330 Hz) to E5 (fundamental frequency≈660 Hz) and normalized to floating point values in [0,1], and volume was mapped exponentially from a 40 dB range to [0,1]. The output of the network is a vector of length 256, where each component represents a mu-law encoded sample value. For training, the audio target signal is coded one-hot, and for generation, the maximally activated unit identifies the audio sample produced by the network.

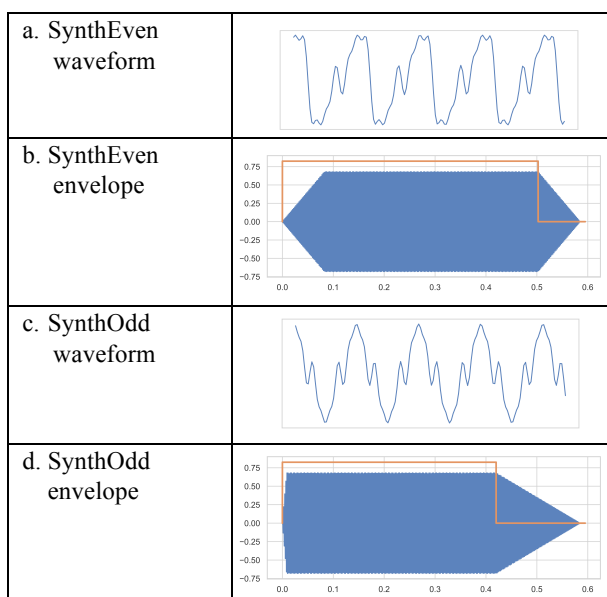




**Figure 1.** The network consists of 4 layers of 40 GRU units each. A four-dimensional vector is passed through a linear layer as input and the output is a one-hot encoded audio sample.

## 2.2 Training Data

Training data consists of 2 instruments, each with 13 different pitches and 25 different volume levels. The two instruments each have their own harmonic structure and transient durations. “SynthEven” is constructed of even harmonics only, and “SynthOdd” with odd harmonics only. The waveforms can be seen in Figure 2a,c.



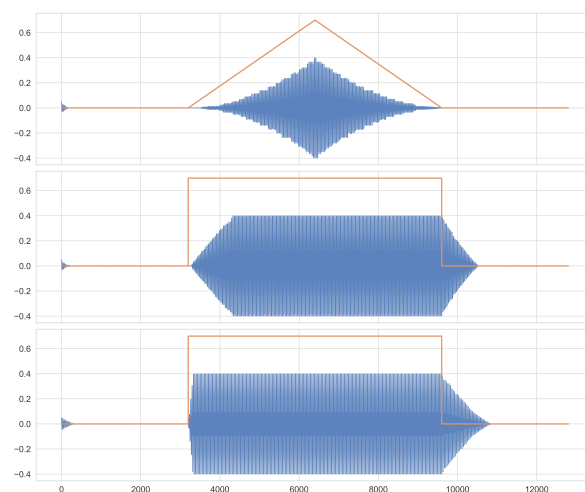
**Figure 2.** Characteristics of the two synthetic instruments used to train the neural network. The synthetic instruments have different wave forms (a,c) and different attack and decay slopes (b,d). The transients follow sudden changes in the volume parameter (orange line).

In addition, SynthEven is constructed with attack and decay transients with a linear slope of  $\pm 10$  volume units/sec, while SynthOdd has an attack with slope  $+100$  and a decay with slope  $-5$  volume units/sec. (Figure 2b, d). Note that transients all have constant slope which means that the duration of the transients depends on the steady-state volume of the tones.

## 3. RESULTS

### 3.1 Steady-state volume

The focus of the current work is on the transient responses to sudden changes in the volume parameter. However, for musical playability, we still require the trained instruments to track the volume parameter at steady state as well as over smooth changes across its range. Figure 3 shows the output of the network for the two instruments in response to various input volume parameter patterns.



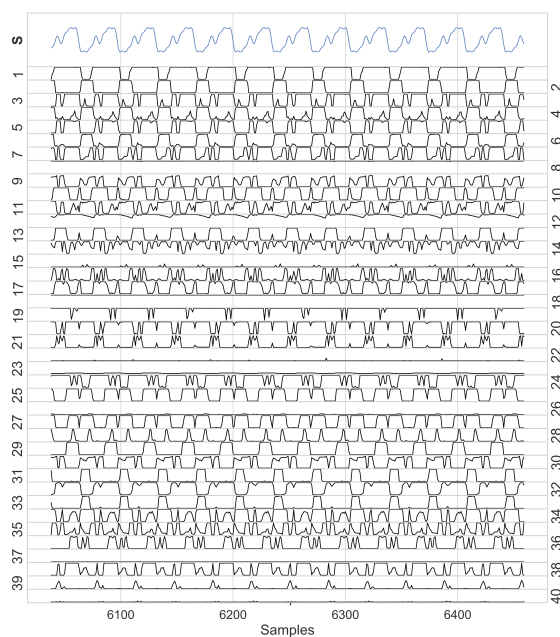
**Figure 3.** From top to bottom: a) SynthEven, pitch=0.5 (B  $\flat$  4,  $\sim 466$  Hz), response to smooth volume change over 400 ms. b) SynthEven, pitch=0.5, response to sudden volume parameter changes c) SynthOdd, pitch=0.5, response to sudden volume parameter changes. In all three scenarios the volume parameter (orange line) was adjusted between a minimum of 0 and a maximum of 0.7. The x-axis depicts sample number.

The network is thus capable of learning to respond to changes in the conditioning input with a state-dependent response extended in time. Furthermore, since the network is trained on two different instruments with different transient characteristics, the temporal response depends on a second conditioning parameter specifying the instrument.

## 4. HIDDEN LAYER PATTERNS

Next, we take a closer look at the hidden unit activation patterns during synthesis to understand the network computations. With only 40 nodes per layer, we can visualize the entire network activation patterns over time.

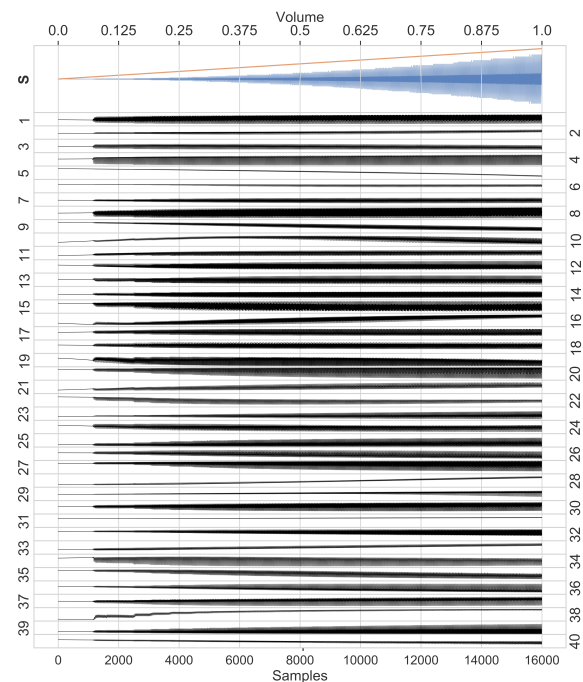
Visualizations show that almost all nodes oscillate with individual characteristic waveforms as illustrated in Figure 4. That waveform is similar across different conditioning parameters, except that the periodicity of the waveforms tracks the period of the pitch specified by the pitch parameter. The node activations show almost none of the frequency selectivity we find in hair cells and neurons along the hearing pathways in animal and human brains. This was somewhat unexpected since a distribution of frequency-specific response patterns have been found in other types of learning networks that operate on audio data that learn “efficient” representations for diverse audio training data (e.g. Lewicki [4]; Hoshen et al. [5]; Sailor and Patil [6]). For the scope of this paper, we just note the pitch-locked oscillatory pattern, but focus on responses to volume input.



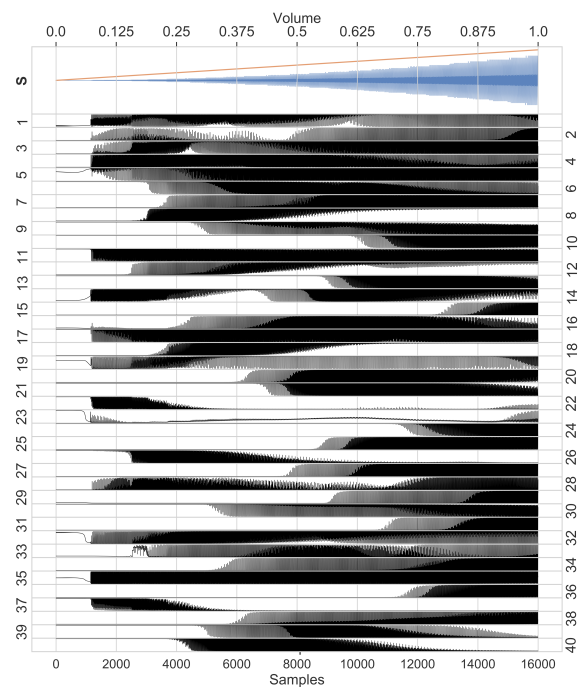
**Figure 4.** The 40 hidden unit responses in Layer 4 of a section of generated audio labelled **S** spanning approximately 400 samples. Each hidden unit displays a characteristic response waveform, the shape of which changes in response to volume level and instrument. In contrast, varying the pitch parameter changes the period of the hidden waveform without altering its overall pattern.

The four hidden layers in the network show distinctly different patterns in response to volume changes. Layer 1 (the layer closest to the input) seen in Figure 5, consists of units most of which oscillate with amplitudes correlated with the volume parameter. A few have a “DC offset” (some positive, some negative) that also tracks the volume parameter. However, none show volume-specific selectivity in their response patterns.

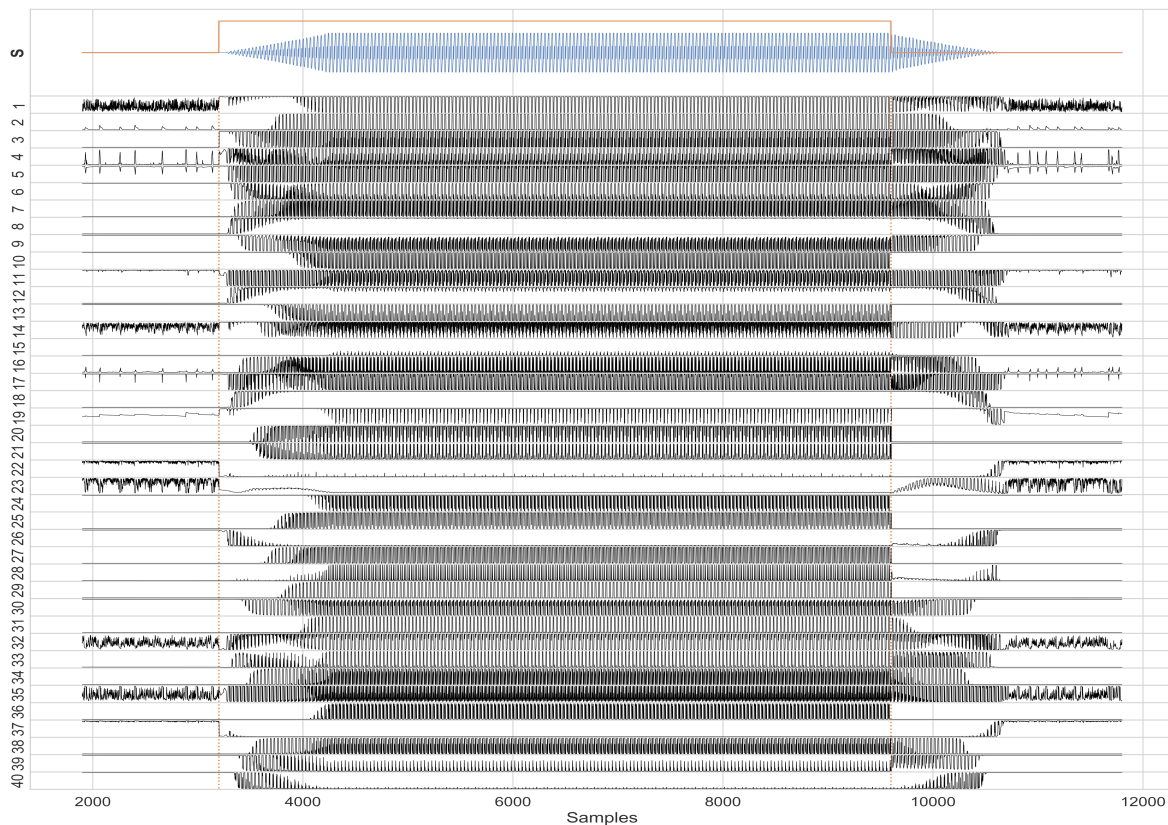
Each succeeding deeper layer shows more complex structure. At the last hidden layer (prior to the linear layer connecting them to the output units), patterns of volume selectivity are clearly visible (Figure 6).



**Figure 5.** Layer 1 (shallowest) hidden unit responses to a continuous increase in volume. All nodes oscillate with amplitude that correlates with the input parameter (as well as with the amplitude of the output signal).



**Figure 6.** Layer 4 (deepest) hidden unit responses to a continuous increase in volume. Individual hidden units clearly show unique volume selectivity.



**Figure 7.** The response of units in the deepest hidden layer to the sudden onset and offset of the volume parameter demarcated by the dotted orange lines, with the synthesized signal and time evolution of the volume parameter shown in the top row. Conditioning parameters used were as follows: instID=SynthEven, pitch=0.5, volume=0 to 0.8 to 0.

Considering only the amplitude of oscillation and not the DC offset, we also see in Figure 6 that node 37, for example, is maximally responsive to low volume; nodes 39 and 40 are responsive to mid-range volumes only, although node 39 has an upwardly-shifted and wider range sensitivity than node 40. Node 15 and 23 only respond to high volumes. We have found no responses sensitive to the direction of slowly changing volume. These patterns are also robust across pitch.

#### 4.1 Transient responses to abrupt volume changes

The responses of hidden neurons to abrupt changes in the volume parameter are more complex, as would be expected, since during the attack and decay transients there is a “mismatch” between the volume parameter and the volume of the output signal. The mismatch is negligible during steady state or the slow sweeping volume changes considered previously.

Figure 7 shows the deepest hidden layer for the response of SynthEven (i.e. trained with symmetrical attack and decay slopes) to a sudden onset and offset of the volume parameter. The output signal *S* (along the top of the figure) is close to the signal used to train this parameter pattern, although further tests showed the overall shape and length of the decay being somewhat inconsistent and

fairly dependent on the parameter combination and the priming signal (a single random sample) used to initialize the synthesis process. Note that the transients in the signal output amplitude are an order of magnitude faster than the volume sweep used in Figure 5 and Figure 6.

One notable feature of this map is that although the output signal amplitude is roughly symmetric following the onset and offset of the volume parameter, the response of the units in this layer are not. Far fewer nodes show an immediate change in activation following the onset of the volume parameter than those that do to the offset of the parameter. Such behavior is reflected in Figure 7 where following the offset of the volume, certain units immediately cease to oscillate (e.g. 10, 13, 20, 25), while others continue to respond with the decaying amplitude.

Some of the patterns of volume selectivity that we found during the slow volume sweep (Figure 6) are still visible during transients in the same units. For example, unit 40 responds to low volume in the output signal during both the attack and decay transients, just as it did during the sweep. Units 9 and 10 maintain their relative volume selectivity during the attack transient as for the sweep. However, unit 10 shuts off immediately with the

volume parameter, while unit 9 is active until the output signal almost disappears.

In general, the attack portion of the signal is more reliably generated and consistent in timing than the decay portion. This might be due to the asymmetry of the training regime. During training, the volume always changes from 0 to the target volume level for the attack, and from the target volume level to 0 prior to the decay. This means that for the portion of the signal following the step change, the attacks portion of the signal was trained concurrent with 25 different volume levels, while the decay portion occurred while the volume was at 0 for all examples, no matter what the steady-state volume prior to the initiation of the decay. A more effective training scheme might be to train on step-ups from non-zero values for attacks, and more importantly, to train on smaller steps down (not all the way to zero) for decays.

## 5. CONCLUSIONS

Wyse [2] developed an RNN that learns to generate signals for different synthetic and natural instruments (Engel et al. [7]) conditioning on pitch and volume so that after training, the models could be played under controls similar to musical instruments. We showed that training examples could be sparse in pitch, and trained only on steady state signals, yet the model responded quickly and accurately to pitch parameter values and sequences it had never seen during training. In the current work, we have shown that the same model can also capture attack and decay transients where the response to the conditioning input is extended over time.

Transients are proving more difficult to capture than pitch or steady-state volume in this small model. They are less robust, more sensitive to priming signals (used to initialize the hidden state) and noise, and do not seem to generalize as easily as pitch or steady-state volume to parameter values and sequences not seen during training. Future work will be necessary to increase the robustness of these results and to apply the network to natural musical instrument data.

We also explored the activation patterns of nodes in hidden layers in response to volume changes and found more selectivity in response to volume levels than was apparent for pitch. This provides a deeper understanding of how the network computes its sound-modeling task, which will help guide the further development of this type of network for learning interactive musical sound synthesis models.

## Acknowledgments

This research was supported by a Singapore MOE Tier 2 grant, “Learning Generative and Parameterized Interactive Sequence Models with Recurrent Neural Networks,” and by an NVIDIA Corporation Academic Programs GPU grant.

## 6. REFERENCES

- [1] J. M. Grey, “Timbre discrimination in musical patterns,” *The Journal of the Acoustical Society of America*, vol. 64, no. 2, pp. 467–472, 1978.
- [2] Wyse, L., “Real-valued parametric conditioning of an RNN for real-time interactive sound synthesis,” 6th International Workshop on Musical Metacreation, International Conference on Computational Creativity (ICCC) June 25–26, Salamanca, Spain, 2018
- [3] K. Cho, B. V. Merriënboer, D. Bahdanau, and Y. Bengio, “On the Properties of Neural Machine Translation: Encoder–Decoder Approaches,” *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014.
- [4] M. S. Lewicki, “Efficient coding of natural sounds,” *Nature Neuroscience*, vol. 5, no. 4, pp. 356–363, 2002.
- [5] Hoshen, Y., Weiss, R. J., & Wilson, K. W. “Speech acoustic modeling from raw multichannel waveforms,” In *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on* (pp. 4624–4628), 2015.
- [6] Sailor, H. B., Patil, H. A., Sailor, H. B., & Patil, H. A. “Novel unsupervised auditory filterbank learning using convolutional RBM for speech recognition,” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(12), 2341–2353, 2016.
- [7] Engel, J., Resnick, C., Roberts, A., Dieleman, S., Eck, D., Simonyan, K., & Norouzi, M. “Neural audio synthesis of musical notes with wavenet autoencoders,” *arXiv preprint arXiv:1704.01279*, 2017.



# Predicting Perceived Dissonance of Piano Chords Using a Chord-Class Invariant CNN and Deep Layered Learning

Juliette Dubois

ENSTA ParisTech  
jdubois@ensta.fr

Anders Elowsson

KTH Royal Institute of Technology  
anderselowsson@gmail.com

Anders Friberg

KTH Royal Institute of Technology  
afriberg@kth.se

## ABSTRACT

This paper presents a convolutional neural network (CNN) able to predict the perceived dissonance of piano chords. Ratings of dissonance for short audio excerpts were combined from two different datasets and groups of listeners. The CNN uses two branches in a directed acyclic graph (DAG). The first branch receives input from a pitch estimation algorithm, restructured into a pitch chroma. The second branch analyses interactions between close partials, known to affect our perception of dissonance and roughness. The analysis is pitch invariant in both branches, facilitated by convolution across log-frequency and octave-wide max-pooling. Ensemble learning was used to improve the accuracy of the predictions. The coefficient of determination ( $R^2$ ) between rating and predictions are close to 0.7 in a cross-validation test of the combined dataset. The system significantly outperforms recent computational models. An ablation study tested the impact of the pitch chroma and partial analysis branches separately, concluding that the deep layered learning approach with a pitch chroma was driving the high performance.

## 1. INTRODUCTION

The concept of dissonance has a long history. However, the experimental study of dissonance dates back only to the middle of the 20th century. Both its definition and causes are still subject to discussion. In early studies [1], consonance is defined as a synonym for beautiful or euphonious. Sethares [2]) proposed a more general definition of dissonant intervals: they sound rough, unpleasant, tense and unresolved.

According to Terhardt [3], musical consonance consists both of sensory consonance and harmony. Similar to this explanation, Parncutt [4] considers the existence of two forms of consonance, one being a result of psychoacoustical factors (sensory consonance) and the other one relating to the musical experience and the cultural environment (musical or cultural consonance). Dissonance can thus be divided into sensory dissonance and musical dissonance. Musical dissonance depends on our expectation and musical culture, which makes it difficult to evaluate [5]. In [6],

several acoustic factors are isolated and the influence of each one is evaluated separately.

Dissonance is closely related to another sensory concept, *roughness*. Roughness applies both for music and natural sounds, and occurs when two frequencies played together produce a beating. According to Vassilakis and Kendall (2010) [7], sensory dissonance is highly correlated to roughness.

Various sensory dissonance models were proposed since the middle of the 20th century. Based on Helmholtz' theory, Plomp and Levelt [1] conducted an experiment using pure sine waves intervals. The result of this study is a set of dissonance curves for a range of frequencies and for different frequency differences. Sethares [8] gives a parametrization of these curves, resulting in a computational model of dissonance in several cases: for two or several sine waves of different amplitudes, for one complex tone, for two notes from the same timbre.

Vassilakis developed a more precise model which estimates the contribution depending on the partial amplitudes and the amplitude fluctuation [9]. This computational model, including a specific signal processing method for extracting the partials, is available online [10].

These models are closely connected, and revolve around the same core concept of bandwidth and proximity of partial frequencies. Other approaches have also been suggested: see Zwicker and Fastl [11], or Kameoka and Kuriyagawa [12]

Schön [13] conducted an experiment in which listeners rated the dissonance of a range of chords played on a piano. Dyads (chords with two notes) and triads (chords with three notes) were played and listeners were asked to rate the dissonance for each chord. The experiment showed that dissonance is easy to rate with a rather high agreement and thus could be considered as a relevant feature for describing music from a perceptual point of view [14]. The rating data from [13] together with the data from [15] were used in the current model.

Perceptual features of music, such as perceived dissonance, *speed*, *pulse clarity*, and *performed dynamics* have received an increasing interest in recent years. They have been studied both as a group ([14, 16, 17]) and through dedicated models ([18–20]). The trend has been towards data driven architectures, foregoing the feature extraction step. Another trend is that of "deep layered learning" models in MIR, as defined by Elowsson [21]. Such models use an intermediate target to extract a representation accounting for the inherent organization of music. The strategy has been

Copyright: © 2019 Juliette Dubois et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

applied for, e.g., beat tracking [22] and instrument recognition [23] in the past. In this study, we show how pitch estimates from a previous machine learning model can be reshaped and fed as input for predicting dissonance.

### 1.1 Overview of the article

In Section 2, we describe the two datasets and the process for merging them. Section 3 is focused on the input representation, detailing how it was extracted for each of the two branches of the CNN. In Section 4, the design of the CNN is described, as well as the methodology used for ensemble learning and the parameter search. Section 5 presents the evaluations methodology and the results. It also includes a small ablation study and a comparison with a previous model of dissonance. Section 6 offers conclusions and a discussion of the results.

## 2. DATASETS

A total of 390 chords were gathered, coming from two different experiments [13, 15]. In these two experiments, the listeners were asked to rate the dissonance of recorded piano chords. A definition of dissonance was given to the listeners. In [15], the consonance was defined as “the musical pleasantness or attractiveness of a sound”. In particular, “if a sound is relatively unpleasant or unattractive, it is referred to as dissonant”. In [13], the following definition was given: “dissonant intervals are often described as rough, unpleasant, tense and unresolved”. Both definitions referred to the unpleasantness of a sound. However, the description from [13] was more precise and already includes the fact that only intervals were studied. The definitions were close enough to give reason to believe that the same feature was evaluated.

### 2.1 First dataset

The first dataset ( $D_1$ ) comes from the experiment conducted in [13]. It contains 92 samples of 0.5 second each, created with a sampled piano in Ableton Live. Two kinds of chords were played, consisting of either two notes (dyads) or three notes (triads). The dyads were either centered around middle C or one fifth above, ranging from unison to a major tenth. The same process was used for the triads. In total, the dataset contained 34 dyads and 58 triads.

Thirty-two listeners were asked to evaluate the sound from “not dissonant at all” to “completely dissonant”, using a web interface. The listeners’ musical background varied but were mostly on an amateur level with an average practice time of 4 hours per week. The inter-rater reliability as estimated by Cronbach’s alpha was 0.95.

### 2.2 Second dataset

The second dataset ( $D_2$ ) contains 298 sound examples [15]. Each sound example was recorded from a piano and has a length of approximately 2 seconds. In this dataset, there are 12 dyads, 66 triads and 220 tetrads (chords with four notes). The frequencies were adjusted so that the mean of the fundamental frequencies was middle C (263 Hz). The

pitches follow a just intonation ratio, differing from the standard equal-tempered tuning used in  $D_1$ . Thirty musically trained and untrained listeners from Vienna and Singapore rated all the examples. The inter-rater reliability as estimated by the average intraclass correlation coefficient (ICC) ranged from 0.96 to 0.99. The ratings were averaged across all listeners.

### 2.3 Merged dataset

We used the average listener rating of dissonance of each chord as a target for our experiment. In  $D_1$ , the dissonance ranged from 0 to 40, 40 being the most dissonant. In  $D_2$ , the dissonance ranged from 1 to 4, 1 being the most dissonant. Therefore, the ratings of the latter dataset were first inverted by multiplication with -1. The listener ratings ( $A$ ) were then normalized according to

$$A_{normalized} = \frac{A - \min(A)}{\max(A) - \min(A)}. \quad (1)$$

After normalization, the most consonant chord had a rating of 0 and the most dissonant chord had a rating of 1. The input data were also normalized. (See Section 3).

## 3. NETWORK INPUT

Two input representations were extracted from the audio files: the constant-Q transform (CQT) spectrum and a pitch chroma. These representations aim to catch different aspects of the audio file. The representation from the CQT can capture spectral aspects of the audio, such as the distance between partials, whereas the pitch chroma represents the audio at a higher level corresponding (ideally) to the actual notes that were played.

### 3.1 Pitch chroma

A pitch chroma was extracted from a Pitchogram representation, as illustrated in Fig. 1. To extract the pitch chroma, the implementation from [24] was applied to the audio files for first extracting a Pitchogram representation. This representation has a resolution of 1 cent/bin across pitch and a frame length of 5.8 ms. The Pitchogram was thresholded at 2 to remove lower noisy traces of pitch. Then, a Hanning window of width 141 was used to filter across pitch. In our initial model, we then extracted activations across pitch at semitone-spaced intervals (12 bins/octave). This gave unsatisfying results, presumably due to the out-of-sync spacing with the just intonation ratios chords in  $D_2$ . Therefore, pitch activations were instead extracted in intervals of 25 cents (48 bins/octave), ranging between MIDI pitch 25 and 103. The mean activation across time for each pitch bin was then computed, using activations from time frames 20-70. The output of this filtering will be referred to as the *pitch vector*.

A pitch chroma vector, ranging an octave, was computed from the pitch vector by taking the average activation wrapped across octaves. Three chroma vectors were stacked across pitch as shown in Fig. 1. The top 6 semitones and bottom 6 semitones were then removed. This stacked pitch

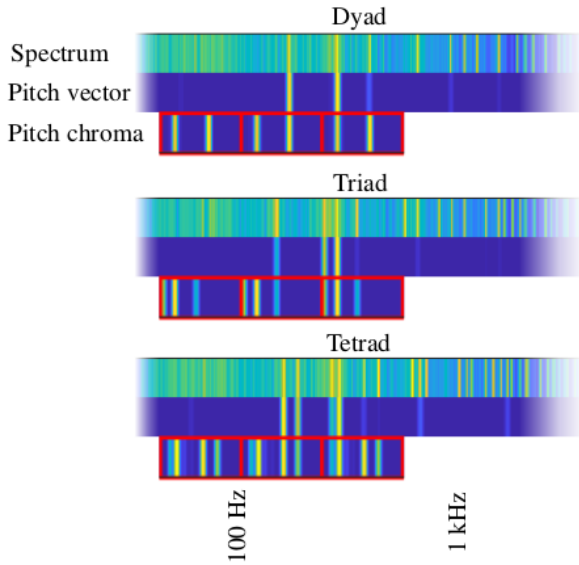


Figure 1: The log-frequency spectrum, computed pitch vector, and pitch chroma for three chords from dataset  $D_2$ . The chords are a dyad, a triad, and a tetrad. As shown, the pitch tracking preprocessing accurately identify the fundamental frequencies. The three stacked pitch chroma octaves are indicated with red rectangles.

chroma vector of size  $[1 \times 96]$  will be referred to as the *pitch chroma* in this paper.

### 3.2 Normalization of the inputs

For each music example  $i$  the pitch chroma and the CQT were then normalized, using the same formula as in Section 2.3:

$$A_{i,normalized} = \frac{A_i - \min(A_i)}{\max(A_i) - \min(A_i)}. \quad (2)$$

### 3.3 CQT vector

To extract a spectral input representation, the recordings were processed with the built-in MATLAB function `cqt`, which uses nonstationary Gabor frames (see [25], [26]). This produced a spectrogram representation with logarithmically spaced frequency bins. We used 60 bins per octave, and the range of the CQT is six octaves, 80 Hz - 5.1 kHz. The mean magnitude across time was then computed for each frequency bin, using only the first half of each audio file (the second half of the audio file has a lesser contribution from higher harmonics). The last preprocessing stage was to compute the log magnitude of this mean:

$$CQT_{mean} = 20 \log_{10}(CQT). \quad (3)$$

The resulting vector contains 360 values.

## 4. NETWORK DESIGN

### 4.1 Architecture

The architecture of the network is shown in Fig. 2.

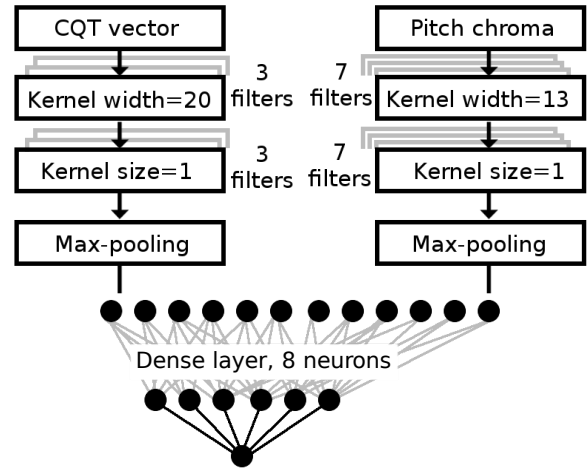


Figure 2: Architecture of the neural network

As shown, the CNN is a directed acyclic graph divided into two branches, where one branch processes the input from the CQT and the other branch processes the pitch chroma representation.

The first layer in the CQT-branch was a convolutional layer of size  $[20 \times 3]$ . The kernel was designed to have a width that covers at least two close partial peaks in the spectrum. The aim of the layer is to capture the interactions between peaks that are adjacent across frequency, as this should convey roughness as outlined in Section 1.

The subsequent convolutional layer also had [3] filters, but each filter had a width of one, therefore only extending across depth. A max-pooling filter was then applied across frequency to capture the most relevant partial interaction in different bands. The max-pooling filter with a width 60 and no stride was then applied.

The branch processing the pitch chroma also had two convolutional layers. The first layer operating across pitch had a size of  $[49 \times 7]$ . Since the edges were not padded during processing, the processing shrinks the pitch chroma to a width of 48, corresponding to pitches within the same octave. This was followed by a layer of width 1 that extended across depth. A max-pooling filter of width 48 (the full range) was then applied to each filter output. Up until this processing stage, the chroma branch of the CNN has operated in a pitch class equivariant way – the same operation has been applied to all pitch classes, with the pitch class displacement intact across chroma (hence *equivariance* and not *invariance*). Through the pooling operation across chroma, the *pitch class equivariance* is transformed into *pitch class invariance*. However, since the activations after pooling will relate only to the intervals of concurrent tones, the system is best defined as having a *chord class invariant* architecture. After max-pooling, the branches are concatenated (see Fig 2) and passed to a dense layer with 8 neurons. The output layer consisted of a single neuron.

The activation functions after the CQT and the dense layer were rectified linear units (ReLUs), and the activation func-

tions in the whole chroma branch were leaky ReLUs:

$$f(x) = \begin{cases} x & x \geq 0 \\ 0.2x & x < 0 \end{cases} \quad (4)$$

The network was trained for 40 epochs, using the RMS propagation optimizer and the mean square error as a loss function.

## 4.2 Ensemble learning

We used ensemble learning, training multiple instances of each network and averaging their predictions. A total of five models were employed in the ensemble, all using the same architecture but with varying random initialization of their parameters. The random initialization of neural networks will decorrelate the errors of the various models [27]. The average prediction from the different models can then be expected to provide better estimates than when randomly choosing one of them [28]. A similar strategy has been used before for training a model to predict perceived performed dynamics in music [20].

## 4.3 Parameter search

A wide range of possible settings was tried with a parameter sweep. For each setting, a model was trained and evaluated with 5-fold cross-validation.

The explored parameters, with tested parameter variations in parenthesis, were: the size of the kernel for the filter operating on the CQT (20 - 10 - 30), the number of filters for the first convolutional layer operating on the CQT and pitch chroma (6 - 7 - 8 for the CQT and 2 - 3 - 4 for the pitch chroma), the pooling size for the CQT branch, and the number of neurons in the dense layer (7 - 8 - 9). The pooling size of 60 was chosen, which corresponds to the range of an octave.

We also tried to include an additional feature inserted at the dense layer, which indicated the dataset of each chord example (1 or 2). This feature did not improve the performance of the network, and thus it was not kept.

# 5. RESULTS

## 5.1 Train and Test conditions

As there were only few data available, the system could easily overfit. To compensate for this lack of data, cross-validation was implemented, and the two datasets were also combined. These two techniques aim at adding variety in the learning set.

Thus, different methods were used to evaluate the performance of the network:

- $A$  – Cross-validation on both datasets combined.
- $B_1$  – Cross-validation within dataset  $D_1$ .
- $B_2$  – Cross-validation within dataset  $D_2$ .
- $C$  – Train on dataset  $D_2$  and test on dataset  $D_1$ .

We used 10-fold cross-validation for  $A$ ,  $B_1$  and  $B_2$ , splitting the datasets into ten folds (nine folds for training and one fold for validation).

For the evaluation  $C$ , the system was trained with the dataset  $D_2$  and evaluated on the dataset  $D_1$ . Given that the two datasets have rather different characteristics (timbre, tuning, and number of notes in the chords) this evaluation condition is more challenging. Since the dataset  $D_1$  consists of so few examples, the opposite evaluation condition (train on  $D_1$ , evaluate on  $D_2$ ) was not explored.

The metric used to compare the prediction and the rated dissonance was the coefficient of determination,  $R^2$ , computed as the squared Pearson correlation coefficient, including an intercept.

Confidence intervals (95 %) were computed, based on the variation in results between different test runs. The 5 test runs were then sampled with replacement 10000 times and the distribution of mean correlations calculated.

## 5.2 Main Results

The coefficient of determination  $R^2$  across the different test conditions ( $A$ ,  $B_1$ ,  $B_2$ , and  $C$ ) are shown in Table 1.

Test condition	Average $R^2$	95 % CI
$A$	0.631	0.622 - 0.634
$B_1$	0.612	0.590 - 0.634
$B_2$	0.644	0.621 - 0.665
$C$	0.583	0.561 - 0.600

Table 1: Coefficient of determination  $R^2$  for the different test conditions, including 95% confidence intervals computed across the different test runs.

The predicted dissonance with respect to the target value for each music example is plotted in Fig. 3. Each point in this figure corresponds to the value of dissonance for one music example. The x-coordinate of the point is the target value of dissonance and the y-coordinate is the prediction of dissonance. For each test condition, the predictions of five different test runs (respectively called prediction 1 - 5 in the figure) are shown.

The cross-fold validation for each dataset gives comparably good results. The test on  $D_2$  gives better results, which could be explained by the number of sample in each dataset:  $D_2$  has four times more sample than  $D_1$ .

The cross-fold validation when combining both datasets yields better results than cross-fold validation for every single dataset. When combining both datasets, the network has a few more examples to train on. This also adds a lot a variety in the training sets, given all the differences listed before. The size of  $D_2$  added with  $D_1$  is not very different than the size of  $D_2$ , but the performance in  $A$  is significantly better than in  $B_2$ . This may indicate that by adding variety in the training set, the network learns much better.

The test  $C_1$  is the only test in which the network learns on only one dataset, and this configuration gives the worse performance. Presumably, the network overfits on  $D_2$  and cannot generalize well enough in order to predict better the values from  $D_1$ .



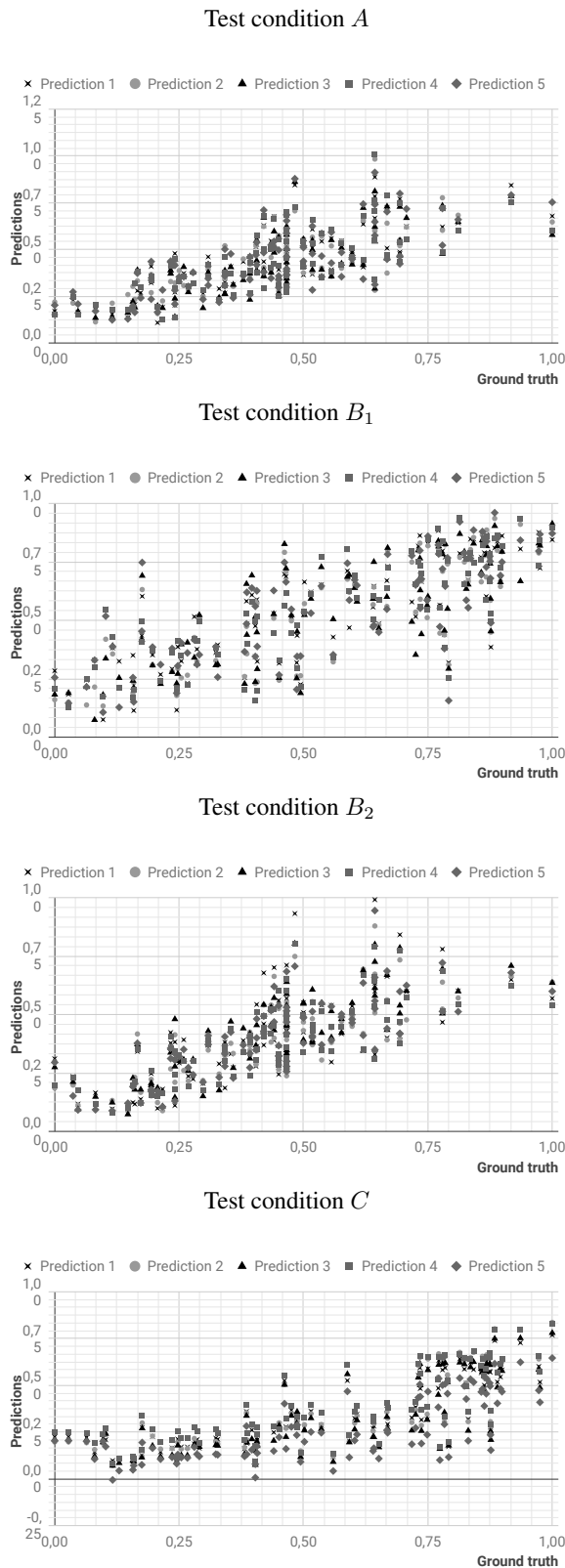


Figure 3: Predictions in relation to ground truth annotations for the different test conditions

Test Condition	Average $R^2$	95 % CI
Network using only the pitch chroma		
$A$	0.618	0.610 - 0.624
$B_1$	0.641	0.629 - 0.649
$B_2$	0.628	0.620 - 0.633
$C$	0.607	0.589 - 0.625
Network using only the CQT vector		
$A$	0.417	0.409 - 0.426
$B_1$	0.304	0.281 - 0.327
$B_2$	0.460	0.439 - 0.484
$C$	0.180	0.139 - 0.221

Table 2: Coefficient of determination  $R^2$  for the network with only one input: CQT or pitch chroma, including 95% confidence intervals computed across the different test runs.

### 5.3 Contribution from each input source and branch

In order to evaluate the importance of the pitch chroma and CQT vector for performance, we also ran the full experiment using only the pitch chroma in a single branch or only the CQT vector in a single branch. Other than this, the same settings were used during training, and the same metric used for testing. The results are shown in Table 2.

Using pitch chroma as the only input consistently gave better results than when only using the spectral input from the CQT. Furthermore, the pitch chroma had better results than the combined main model (reaching outside of the 95 % CIs) for test conditions  $B_1$  and  $C$ . The CQT vector had particularly low results for test condition  $C$ . This condition tests the ability of the architecture to generalize since the system is trained on one dataset and tested on another with, presumably, different characteristics pertaining to, e.g., timbre. The results confirms that the deep-layered learning approach to MIR [21], in this paper using transfer learning of equivariant feature maps, can yield significantly better results than end-to-end learning for small datasets.

### 5.4 Comparison with a Computational Model

In this section, comparison of the performances of the system with a state of the art model is presented. The most recent model was proposed by Vassilakis [9] and does not use machine learning. It was already implemented as the function `mirroughness` in the MIR toolbox [29], which is the implementation we used for the comparison.

With this function, a dissonance value is given for each unit of time, which was not directly comparable with the single value given by the listeners. Considering that a human listener would not take the length of the recording into account, we chose to take the mean of the five highest values.

With this method, a dissonance value was computed for each music example. The squared correlation coefficient was then computed between this dissonance and the target dissonance. The results are shown in Table 3.

The  $R^2$  score obtained here is lower than the performances obtained in the article presenting the model [9]. This could be explained by the fact that the computational model was tested and adapted to synthetic sine waves, whereas the audio files in this experiment came from a sampled piano. The timbre of the piano presumably increases the complexity of the sound and reduce the accuracy of the model.

Dataset	$R^2$
$D_1$	0.17
$D_2$	0.34

Table 3: Coefficient of determination  $R^2$  for the computational model for each dataset

## 6. CONCLUSION AND DISCUSSION

A model using a convolutional neural network was developed for predicting the dissonance in recordings of piano chords. The model achieved better results than previous computational models, even though there were few samples in the datasets.

The two datasets differ in at least four ways:

- The chords were played with two different piano models, producing differences in, e.g., timbre.
- One dataset was performed with equal-tempered chords, and one with just intonation ratio. The model, therefore, had to handle micro-tuning deviations and how they affect dissonance.
- One main difference is the polyphony level of the chords:  $D_1$  has no tetrads whereas these constitute more than two-thirds of  $D_2$ .
- The two datasets were rated by two different groups of listeners. Therefore, it can be expected that random variations between preferences in the two groups gave annotations that varied in complex ways.

We conclude that the tests validate the potential of intermediate targets accounting for the inherent organization of music. The "deep layered learning" approach [21] using only the pitch chroma branch gave significantly better results than when using only the spectral CQT vector branch. In particular, a comparison between the results for test condition  $B_2$  and  $C$  underlines the pitch chroma-only model's high generalization capability. The  $R^2$  only fell slightly ( $0.628 - 0.607 = 0.021$ ) when testing on an unseen dataset instead of using cross-validation. For the main model with both branches, the results fell more between these test conditions ( $0.644 - 0.583 = 0.061$ ).

During development, we tested a few different architectures, with fewer learnable parameters in total, but those architectures gave lower results. It seems like the architecture allowed for a fairly high amount of learnable parameters in relation to the low number of ground truth data points.

In future work, a wider range of architectures could be tried, reflecting insights gained from the small ablation study. The pitch chroma branch can be designed as the only branch, exploring improvements related to, e.g., pitch resolution, depth, and pooling. A related study [30] has showed that it is possible to compute several chroma within the network instead of as a preprocessing step, each chroma focusing on different octaves. That study also indicated that average pooling across octaves for key-class invariance can give better results than max-pooling. It could be useful to analyze the weights in the kernel on the trained network. This could make it easier to understand the characteristics selected by the network. The network could also be trained with a much bigger dataset, using several repetitions of the same chord class or using chords from higher and lower octaves.

## Acknowledgments

The authors appreciate the help of Daniel L. Bowling who made the data of his work available.

## 7. REFERENCES

- [1] R. Plomp and W. J. M. Levelt, "Tonal consonance and critical bandwidth," *The Journal of the Acoustical Society of America*, vol. 38, no. 4, pp. 548–560, 1965. [Online]. Available: <https://doi.org/10.1121/1.1909741>
- [2] W. A. Sethares, "Relating tuning and timbre," 1992. [Online]. Available: <https://web.archive.org/web/20100610092432/http://ecserv0.ece.wisc.edu/~sethares/consemi.html>
- [3] E. Terhardt, "The concept of musical consonance: A link between music and psychoacoustics," *Music Perception: An Interdisciplinary Journal*, vol. 1, no. 3, pp. 276–295, 1984. [Online]. Available: <http://mp.ucpress.edu/content/1/3/276>
- [4] R. Parncutt, *Harmony : A Psychoacoustical Approach*. Springer, 1989.
- [5] A. Porres, "Dissonance model toolbox in pure data," in *Pure Data Convention Weimar - Berlin*, 2011.
- [6] J. Mcdermott, A. J. Lehr, and A. J. Oxenham, "Individual differences reveal the basis of consonance," vol. 20, pp. 1035–41, 06 2010.
- [7] *Psychoacoustic and cognitive aspects of auditory roughness: definitions, models, and applications*, vol. 7527, 2010. [Online]. Available: <https://doi.org/10.1117/12.845457>
- [8] W. Sethares, "Local consonance and the relationship between timbre and scale," *The Journal of the Acoustical Society of America*, vol. 94, 09 1993.
- [9] P. N. Vassilakis, "Perceptual and physical properties of amplitude fluctuation and their musical significance," Ph.D. dissertation, University of California, Los Angeles, 2001.

- [10] “Spectral and roughness analysis of sound signals,” <http://musicalgorithms.ewu.edu/algorithms/roughness.html>. [Online]. Available: <http://musicalgorithms.ewu.edu/algorithms/roughness.html>
- [11] E. Z. H. Fastl, *Psychoacoustics*. Springer, 2009.
- [12] A. Kameoka and M. Kuriyagawa, “Consonance theory part ii: Consonance of complex tones and its calculation method,” *The Journal of the Acoustical Society of America*, vol. 45, no. 6, pp. 1460–1469, 1969. [Online]. Available: <https://doi.org/10.1121/1.1911624>
- [13] R. Schön, “Vertical dissonance as an alternative harmonic-related perceptual feature,” KTH DM 2906 Individual Course in Media Technology, Tech. Rep., 2017.
- [14] A. Friberg, E. Schoonderwaldt, A. Hedblad, M. Fabiani, and A. Elowsson, “Using listener-based perceptual features as intermediate representations in music information retrieval,” *The Journal of the Acoustical Society of America*, vol. 136, no. 4, pp. 1951–1963, 2014.
- [15] D. L. Bowling, D. Purves, and K. Z. Gill, “Vocal similarity predicts the relative attraction of musical chords,” *Proceedings of the National Academy of Sciences*, vol. 115, no. 1, pp. 216–221, 2018. [Online]. Available: <http://www.pnas.org/content/115/1/216>
- [16] A. Friberg and A. Hedblad, “A comparison of perceptual ratings and computed audio features,” in *Proceedings of the SMC 2011 - 8th Sound and Music Computing Conference*, 2011, pp. 122–127. [Online]. Available: [http://www.smc-conference.org/smc11/papers/smc2011\\_163.pdf](http://www.smc-conference.org/smc11/papers/smc2011_163.pdf)
- [17] A. Aljanaki and M. Soleymani, “A data-driven approach to mid-level perceptual musical feature modeling,” *CoRR*, vol. abs/1806.04903, 2018. [Online]. Available: <http://arxiv.org/abs/1806.04903>
- [18] F. A. M. G. . P. J. Elowsson, A., “Modelling the speed of music using features from harmonic/percussive separated audio,” in *14th International Society for Music Information Retrieval Conference (ISMIR 2013)*, 2013, pp. 481–486.
- [19] O. Lartillot, T. Eerola, P. Toiviainen, and J. Fornari, “Multi-feature modeling of pulse clarity: Design, validation, and optimization,” in *In Proceedings of the International Symposium on Music Information Retrieval*, 2008.
- [20] A. Elowsson and A. Friberg, “Predicting the perception of performed dynamics in music audio with ensemble learning,” *The Journal of the Acoustical Society of America*, vol. 141, no. 3, pp. 2224–2242, 2017. [Online]. Available: <https://doi.org/10.1121/1.4978245>
- [21] A. Elowsson, “Deep layered learning in MIR,” *CoRR*, vol. abs/1804.07297, 2018. [Online]. Available: <http://arxiv.org/abs/1804.07297>
- [22] M. D. F. Krebs, S. Böck and G. Widmer, “Downbeat tracking using beat synchronous features with recurrent neural networks,” in *ISMIR*, 2016, pp. 129–135.
- [23] Y.-N. Hung and Y.-H. Yang, “Frame-level instrument recognition by timbre and pitch,” in *ISMIR*, 2018, pp. 135–142.
- [24] A. Elowsson, “Polyphonic pitch tracking with deep layered learning,” *CoRR*, vol. abs/1804.02918, 2018. [Online]. Available: <http://arxiv.org/abs/1804.02918>
- [25] *Matlab documentation on cqt*, <https://se.mathworks.com/help/wavelet/gs/non-stationary-gabor-frames.html>. [Online]. Available: <https://se.mathworks.com/help/wavelet/gs/non-stationary-gabor-frames.html>
- [26] C. Schörkhuber, A. Klapuri, N. Holighaus, and M. Drfler, “A matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution,” in *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*, Jan 2014. [Online]. Available: <http://www.aes.org/e-lib/browse.cfm?elib=17112>
- [27] L. K. Hansen and P. Salamon, “Neural network ensembles,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, Oct 1990.
- [28] R. Polikar, “Ensemble based systems in decision making,” *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–45, Third 2006.
- [29] O. Lartillot, P. Toiviainen, and T. Eerola, “A matlab toolbox for music information retrieval,” in *Data Analysis, Machine Learning and Applications*, C. Preisach, H. Burkhardt, L. Schmidt-Thieme, and R. Decker, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 261–268.
- [30] A. Elowsson and A. Friberg, “Modeling music modality with a key-class invariant pitch chroma cnn,” manuscript submitted for publication, 2019.

# Belief Propagation algorithm for Automatic Chord Estimation

**Vincent P. Martin**  
LaBRI, UMR 5800

Univ. Bordeaux, FRANCE  
vincent.martin@labri.fr

**Sylvain Reynal**  
ETIS, UMR 8051 Univ. Paris Seine

Univ. Cergy-Pontoise, ENSEA, CNRS, FRANCE  
reynal@ensea.fr

**Dogac Basaran**  
Ircam Lab

Sorbonne Univ., FRANCE  
dogac.basaran@ircam.fr

**Hélène-Camille Crayencour**  
L2S, UMR 8506

Univ. Paris-Sud, CNRS, FRANCE  
helene.camille.crayencour@gmail.com

## ABSTRACT

This work aims at bridging the gap between two completely distinct research fields: digital communications and Music Information Retrieval. While works in the MIR community have long used algorithms borrowed from speech signal processing, text recognition or image processing, to our knowledge very scarce work based on digital communications algorithms has been produced. This paper specifically targets the use of the Belief Propagation algorithm for the task of Automatic Chord Estimation. This algorithm is of widespread use in iterative decoders for error correcting codes and we show that it offers improved performances in ACE by genuinely incorporating the ability to take constraints between distant parts of the song into account. It certainly represents a promising alternative to traditional MIR graphical models approaches, in particular Hidden Markov Models.

## 1. INTRODUCTION

This paper focuses on Automatic Chord Estimation (ACE), that is estimating a series of chords from an audio file. Among the oldest tasks ever tackled by MIR, it is essentially an inference problem which consists in estimating the chords of a song (hidden variable) given the audio file from which observations are computed (chromas). While initially relying on hand-crafted features, ACE algorithms have progressively incorporated language models and deep learning in recent times [1]. But even with these advances, the performances of existing approaches stagnate, differing only slightly from one another [2–4].

An important limitation of existing work is that in most models, analysis is typically limited to short timescale. This overlooks long-term structural dependency between music events and does not reflect the rich underlying relational structure. For instance, the chord progression is both related to the high-level semantic structure organization (e.g. in a song, all choruses are likely to have a similar chord progression [5]), and to a lower-level metrical structure organization (e.g. chord changes are likely to happen on downbeats [6]). A fundamental question that remains open

is how to model this complex hierarchical relational structure.

The MIR community has explored a handful of approaches to encode long-term structure with short-term analysis. To our knowledge, one of the first to mention harmonic modeling using graphical model is [7], which uses HMM. Another possible scheme is to rely on recurrences in a song to label all music segments of the same type with the exact same chord progression, replacing all identically labeled tatum by their mean chroma [5]. This approach lacks flexibility however since it ignores possible variations between several occurrences of the same structural segment in a piece of music. A more flexible strategy uses Markov Logic Networks [8] in order to model long-term dependencies between chords, but it is limited by a slow inference process, making it difficult to process long pieces and model complex dependencies. More recently Recursive Neural Networks [9] have been considered since they can, in principle, model arbitrarily complex long-term temporal dependencies. However, they have exhibited difficulties to make the model learn long-term dependencies from data [10] and they do not explicitly use the structure but fuzzy information specified by the network. Finally in [11] the strategy elaborated bears some resemblance to ours, namely a graph is designed so that each chord has a short and a long term context. However, the graph construction and the estimation of the chord sequence is not carried out in the same way: where the author use Expectation-Maximisation, we propose a distinct approach, based on the Belief Propagation message-passing algorithm.

These previously mentioned limitations represent an incentive to explore new approaches inspired by other communities, e.g., statistical physics or digital communication, where information is represented by complex graph models and marginalization also represents a difficult challenge [12]. Indeed when computing marginals of probability distributions with a huge number of degrees of freedom, brute force search has an exponential complexity. Numerous algorithms have thus been devised in the last twenty years or so to tackle this issue. Amid those, Belief Propagation (BP) algorithms (also called cluster mean-field algorithms in statistical physics) have emerged as an efficient way to compute marginal probabilities by i) performing iterative updates of local probability distribution (the so-called “beliefs”) based on a sort of local survey of opinions — or gossip — and ii) travelling along the Bayesian graph of constraints to update all beliefs in turns.

In this work we propose to go beyond the current limita-

Copyright: © 2019 Vincent P. Martin et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.



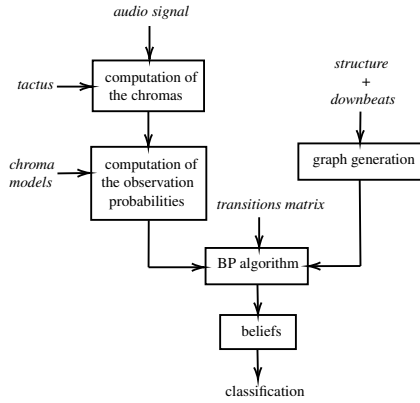


Figure 1. Flowchart of our system. *Italic* represents the the input of the system, and each box corresponds to a part of the signal processing that is detailed in the corresponding section.

tions of ACE by relying on an approach inspired by iterative decoders for error-correcting codes. We take advantage of the BP algorithm to model and incorporate song structure information in the chord estimation process. One of the main benefits of the BP algorithm indeed is that it can embody constraints between any state, whatever their proximity on the timeline. Hence that various long-term correlations can be incorporated in the inference process and make it more robust. In this respect, BP has already been considered as a mean to compute the marginal probability of the state variables in each analysis frame in the case of beat tracking [13] but only correlations between consecutive events were considered. In the present work, we aim at exploring how to encode long-term structure using BP algorithm.

The paper is structured as follows. In Section 2, we provide a brief description of the proposed system and the Belief Propagation algorithm. In Section 3, we present the advantages of using downbeats and the structure to enhance performance. Section 4 presents the dataset, Section 5 proposes a brief study on noise robustness, Section 6 shows the results of the experiment and a discussion. Section 7 discuss some methodology biases. Finally, conclusions and future work are presented in Section 8.

## 2. BELIEF PROPAGATION FOR AUTOMATIC CHORD ESTIMATION

As in the majority of computational models for ACE [14], the system we consider has a two-step architecture composed of a feature extraction step (in our case, handcrafted chroma features) followed by a classification step. This latter step consists in inferring hidden states, i.e., putting labels on chords from a chord dictionary by using information from the observations.

The system flowchart is shown in Figure 1. Elements in *italic* are the system inputs obtained from the ground truth. Given spectral information (chroma) and a model we can estimate the probability of a given chord from a set of observation vectors. We further feed the conditional prob-

ability of a chord given other chords into the estimation process. Based on these inputs a decoding algorithm computes the most probable sequence of chords, during the pattern matching step. In this paper we compare a BP decoder with a Viterbi algorithm that uses Hidden Markov Models (HMM).

### 2.1 Observation and transition probabilities

The observation probabilities are computed from the chroma vectors in the same way as in [6]: each element in the observation vector is the cosine similarity between the chroma and a theoretical template. We compute tatum-synchronous observations, where tatum (the smallest time interval between two successive notes [15]) are quarter notes here. First they are computed with ground truth beats, and then they are estimated (see Section 4.3).

Transition matrices are an important ingredient of the pattern matching step: they allow us to take into account information from other chords, which in turns improves the inference process. As proposed in [16], we use the perceptual transition matrix elaborated in [17]. Results with the "cycle of fifths" transition matrix proposed in [16] are also provided for completeness in Section 6.

### 2.2 HMM vs BP

As the main objective of this work concentrates on the pattern matching step, we devote this section to highlighting the main differences between the HMM with Viterbi inference and the BP algorithms. We show in particular that we can rewrite the HMM algorithm as a particular case of the BP algorithm. Then we examine how we can incorporate structural information to take full advantage of the BP algorithm.

#### 2.2.1 Viterbi with HMM

A HMM is a statistical model that relates the probability vector of a hidden state to observation and transition probabilities. Let  $x_i$  be the chord to be inferred at position  $i$ . The algorithm is described by the following parameters:  $\pi_i$  is the probability that  $x_i$  is the initial state,  $a_{ij}$  is the transition probability from  $x_i$  to  $x_j$  and  $b_i(O)$  is the probability that observation  $O$  is emitted for chord  $x_i$ . In our case, the hidden states  $x_i$  are the chords that we want to infer ( $x_i \in [1, N_D]$ , where  $N_D$  is the size of the chord dictionary), the observations are the chroma and  $a_{ij}$  and  $b_i(O)$  are the transition matrix and the model to compute the observations.

State  $x_0$  is initialized as the column vector  $(\frac{1}{N_D})_{N_D,1}$ ; there is no a priori distribution of the probabilities. Then Viterbi inference is carried out as follows:

$$\forall i, S_i = \arg \max_k \{b_i(O_k) \times a_{i-1,k}\} \quad (1)$$

#### 2.2.2 Belief Propagation

BP is designed to infer hidden states given observations and transition probabilities between them [18]. Yet the BP algorithm is above all an iterative, message-passing algorithm that leverages the topology of the underlying

Bayesian graph to improve estimates. While the Viterbi inference with HMM is done linearly, BP can use any topology, including cycles. Modeling with an HMM is inspired by the chronology of events in the song and infers a given state using information from the previous state on the timeline. In this respect, HMM draws more upon directed Bayesian networks. On the contrary, BP can use context to constraint a given chord to any other part of the song.

Let  $x_i$  be the chord to be inferred at node  $i$ . The BP algorithm relies on the adjacency matrix of the Bayesian graph, the observation vectors  $\phi_i(x_i)$  and a constraint  $\psi_{i,j}(x_i, x_j)$  between nodes  $i$  and  $j$  that renders existing correlations. HMM transitions matrices are thus a particular case of transition matrices between neighbouring nodes.

### 2.2.3 Sum-Product vs Max-Sum versions

Two flavours exist of the BP algorithm, with specific benefits and drawbacks. The Sum-Product algorithm works as follows: for every node  $j$  associated with chord  $x_j$  to be inferred, the incoming message  $m_{i \rightarrow j}(x_j)$  is computed from node  $i$  using the following "survey" equation,

$$m_{i \rightarrow j}(x_j) = \sum_{x_i} \phi_i(x_i) \psi_{i,j}(x_i, x_j) \prod_{p \in N(i), p \neq j} m_{p \rightarrow i}(x_j), \quad (2)$$

where  $N(i)$  is the neighborhood of node  $i$  on the graph. A given message is thus the product of a local observation probability, a constraint and messages coming from the rest of the graph that in effect convey a poll on "what the best estimate of state  $x_j$  should be". As the process is iterative — since every node is considered in turn until convergence is reached — this equation can be seen as progressively aggregating more and more of local beliefs as messages propagate through the graph. The messages are normalized at each iteration so that they sum to one.

When convergence is reached, we calculate each chord probability (the so-called *beliefs*) by using

$$b_i(x_i) = \phi_i(x_i) \prod_{j \in N(i)} m_{j \rightarrow i}(x_i). \quad (3)$$

and infer the hidden states with

$$x_i = \arg \max_k \{b_i(x_k)\}. \quad (4)$$

Figure 2 shows an example of a message-passing step from node 3 to node 2:

$$\begin{aligned} m_{3 \rightarrow 2}(x_j) &= \sum_{x_i} \phi_3(x_i) \psi_{3,2}(x_i, x_j) \prod_{p \in \{4,5\}} m_{p \rightarrow 3}(x_j) \\ &= \sum_{x_i} \phi_3(x_i) \psi_{3,2}(x_i, x_j) m_{4 \rightarrow 3}(x_j) m_{5 \rightarrow 3}(x_j) \end{aligned} \quad (5)$$

The Max-Sum version computes messages according to

$$m_{i \rightarrow j}(x_j) = \max_{x_i} \phi_i(x_i) \psi_{i,j}(x_i, x_j) \prod_{p \in N(i), p \neq j} m_{p \rightarrow i}(x_j) \quad (7)$$

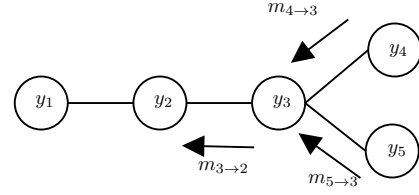


Figure 2. An example of message-passing iteration for the BP algorithm.

It is an interesting, less CPU intensive alternative when one is not interested in the exact marginal probabilities, but only in classification (see [19]), which is our case here. Our results on noise robustness also show that the max-product version provides lower error rates.

### 2.2.4 HMM viewed as a BP algorithm

HMM can be viewed as a very simple BP algorithm where the Bayesian graph is a simple-path, unweighted directed graph going from initial state  $y_0 = S_0 = (\frac{1}{N_D})_{N_D,1}$  to the end of the song, and where messages that are propagated are simply the beliefs, that is,

$$\forall j > i \quad m_{i \rightarrow j}(c) = \phi_j(c) \times \psi_{i,j}(y_{i-1}, c) \quad (8)$$

$$= O_c \times a_{i-1,c}. \quad (9)$$

with the most probable states  $y_j$  being computed at each step by

$$y_j = S_j = \arg \max_k \{m_{j-1,j}(k)\}. \quad (10)$$

## 2.3 Benefits and drawbacks of the BP algorithm

The BP algorithm can easily take into account non-local correlations by using any appropriate  $(i, j)$  edge with specifically tailored constraints.

The method we proposed above may suffer flaws, however. If the graph has a small girth, which might depend on the song content and structure, the algorithm may converge to an incorrect solution, or even not converge at all. This in particular occurs if the set of constraints along a cycle creates conflicting constraints, so that message propagation may lead to beliefs oscillating between two or more chords at each node in the loop.

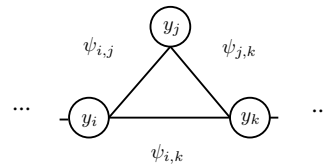


Figure 3. Example of a 3-cycle where the algorithm experiences difficulties converging or does not converge at all.

Populating the Bayesian graph with  $\psi_{i,j}$  constraints can lead to short cycles (at the bar scale) or very large ones (at the large-scale structure level). Short cycles undergo convergence issues (as explained above) but may also converge fairly quickly. On the contrary, large cycles are stable

but consume a lot of iterations to feedback the information, hence they take more time to converge.

A criterion has to be defined to stop the iteration:

$$\forall(i, j) \max_k |m_{i \rightarrow j}^{n+1}(k) - m_{i \rightarrow j}^n(k)| \leq \epsilon \quad (11)$$

We arbitrarily set  $\epsilon = 10^{-12}$  and a maximum number of updates of 200 beyond which message convergence is considered to have failed.

### 3. DOWNBEATS AND STRUCTURE: TWO WAYS TO ENHANCE ACE PERFORMANCES

To take full advantage of the BP algorithm, the next step is to feed the basic linear graph with structural information, namely the downbeats and the structure.

As shown in [8], information contained in the structure of a song improves the performance of ACE. It is incidentally quite intuitive that, when listening to a song and having roughly identified the structure of the song, it is not uncommon to predict the next chords that will be played. BP can leverage this information, creating connections between parts of the song that are very similar, e.g., connecting the first beat of each chorus with the corresponding beat in every other chorus.

Utilizing downbeats proceeds along the same line, yet at the bar scale. As expected by the encouraging results obtained when using downbeats in tonality estimation in [20], using this information in this work has produced encouraging results as well.

#### 3.1 Using downbeats

Inside a bar chords are not independent of each other: songs where chord change every beat are rare and it is not seldom that each chord is repeated twice in a bar. In practice, we connect all the beats of the same bar with a probability  $\psi'$  to be identical. This assumes that almost all chords in the same bar are identical, yet the flexibility of the BP guarantees that the turn-over chords at the end of a bar will not be misinterpreted. From a Bayesian graph perspective, including the downbeats positions allows feeding each node with more mutual information: instead of receiving information from its neighbours only, it receives information from all the other nodes in the bar. The corresponding graph with both downbeats and structural information included is shown Figure 4.

Assuming we retain the aforementioned transition matrix  $\psi$  between subsequent bars (shown in red in Figure 4), there is still one parameter to be determined, i.e., the transition matrix  $\psi'$  between nodes of the same bar (shown in blue in Figure 4). We assume that  $\psi'$  is defined primarily by self transitions, i.e., the probability that the chords are identical, while other probabilities are uniformly distributed:

$$\psi'(i, j) = \begin{cases} \alpha & \text{if } i = j \\ \frac{(1 - \alpha)}{N_D - 1} & \text{else} \end{cases} \quad (12)$$

To set  $\alpha$ , distinct values have been tested ranging from  $\frac{1}{N_D}$  to 1. Values lower than  $\frac{1}{N_D}$  were not tested: they

would imply that self transition are disadvantaged, which would contradict the assumption that chords are mostly identical in a bar. This would also increase conflicting constraints along some graph cycles and make convergence more difficult. Surprisingly the best results were obtained for a self transition of  $\alpha = 0.05$ : one could have expected indeed that higher values would give better results since they bind events in a bar in a stronger way, and chords of the same bar would have higher probabilities to be identical.

All in all this adds a lot of messages to be computed but still the number of messages updates is lower than in the simple chain BP algorithm: short 3- or 4-cycles are created that converge quite quickly. The weakness of this is that it could also preclude convergence if there is contradiction between observations and incoming messages: we thus assume that a low value for  $\alpha$  produces good results (see Section 2.3 for more details on short cycles).

#### 3.2 Using the song structure and long-term correlations

Incorporating the structure allows for feeding far more information into each node: now they also receive information from all the nodes that share the same "position" in the song. For example, the first node of the first verse is connected to the first node of all other verses (see Figure 4). Determining the transition matrix  $\psi''$  between events that are connected by long-term correlations through the song structure follows the same guidelines as for downbeats: we take the same matrix defined by self transitions while other probabilities are uniformly distributed.

For downbeats we tried several values ranging from  $\frac{1}{N_D}$  to 1 and we obtained the best results for  $\alpha = 0.05$ . This process adds a lot of edges to the graph and a lot more messages thus need to be calculated. As opposed to downbeats, incorporating the structure creates large cycles which also need a lot of updates to converge.

The global graph is represented in Figure 4. Populating the graph with both the structure and the downbeats yields the best results: the quick convergence due to short cycles (downbeats) makes up for large cycles that slow down converge. In only 3 songs of the database convergence was not reached.

#### 3.3 Leveraging similarities

An alternative idea is to change the previous  $\alpha$  in  $\psi'$  and  $\psi''$  according to the correlations between any pair of chroma. Indeed, instead of having a tunable parameter that is the same for the whole graph, the similarity constraint varies depending on the similarity between nodes. We compute the self-similarity matrix  $M(i, j)$  (see [21]) and calculate the messages according to

$$\psi'(i, j) = \begin{cases} M(i, j) & \text{if } i = j \\ \frac{(1 - M(i, j))}{N_D - 1} & \text{else} \end{cases} \quad (13)$$

$$\psi''(i, j) = \psi'(i, j) \quad (14)$$

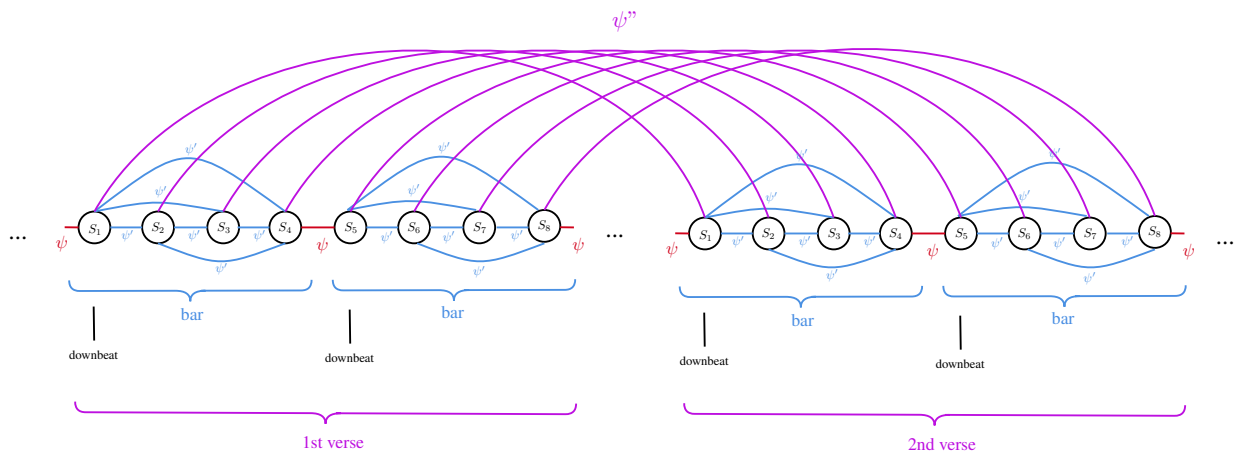


Figure 4. How the structure and the downbeats influence the topology of the Bayesian graph: constraints between chords of a bar are shown in blue ; those between chords of the same structural element are in magenta ; finally, transitions probability between bars are in red.

This strategy yields results with the same quality as with the previous model, yet with longer computation time.

Similarity can also spark new ways to populate the graph. Instead of relying on the ground truth downbeats and structure, we introduce another method based uniquely on similarity. The point is to build a fully connected graph, where each edge is weighted by the degree of similarity between the chroma it connects. Each chord is thus estimated using information from all the other chords of the song, and not just those that share structural position. However, the computation time is quite large as for a graph of size  $N$  tatums, this method requires to compute  $N(N - 1)$  messages.

To keep computation time reasonably low we introduce two parameters:

- $\alpha$  is a similarity threshold. If similarity is lower than  $\alpha$ , we discard the edge between the corresponding nodes.
- $\beta$  is the maximum number of edges connected to a given node.

$\alpha \in 0.9, 0.95, 0.98$  and  $\beta \in 5, 10, 20$  yielded very poor performances. Future works will include investigating this issue.

## 4. REAL WORLD DATASET

### 4.1 Performance estimation

Various methods exist to evaluate the performances of a retrieval task. ACE can be seen as a classification task which requires i) a criterion to tell if the method has reproduced the ground truth to an acceptable degree and ii) classes onto which the different observation can be classified (the so-called dictionary). We invite the reader to refer to [22] for more information on the formalization of Music Information Retrieval.

As in most studies about chord estimation, we consider only the 24 major and minor chords [1]. Chords in the ground truth that are not in the dictionary are projected to

major or minor chords (as in [23]), so that the recall can be computed.

We evaluate the performances of the system with the python library *mir\_eval* [23]. The performance is measured by the Weighted Chord Symbol Recall (WCSR) defined in [24].

### 4.2 Database

The various inference algorithms are tested on the Beatles subset of the Isophonics data set. Following [16], some songs are not considered, due to the uncertainty on their structure or the errors in the ground truth provided by Isophonics. These songs are listed in the following list:

- *Lack of downbeats file*: Get Back, Glass Onion, Revolution 9
- *Incorrect annotations*: Lovely Rita
- *Complicated Metric*: Baby's In Black; You've Got To Hide Your Love Away; Norwegian Wood; She's leaving Home; Long, Long, Long; Oh! Darling; Dig A Pony; Dig It; A taste Of Honey; Lucy In The Sky With Diamonds; Being For The Benefit Of Mr. Kite; Strawberry Fields Forever; All You Need Is Love; Happiness Is A Warm Guy; I Want You (She's So Heavy); Two Of Us; I Me Mine.

We considered 157 songs in our data set. For each song the *wav* audio file, the annotated chords and their respective starting and ending time are available.

### 4.3 Estimated vs ground truth information

To estimate the robustness of our algorithm against variations in the beats and the downbeats, we compute the performance of the system using ground truth beats and downbeats but also using estimated beats and downbeats. These estimated beats and downbeats are processed with state of the art Python library *madmom* [25]. The algorithms contained in this library use Recursive Neural Networks and Deep Bayesian Networks. The only drawback



of this library is that the downbeats can be estimated only with some rhythmic signatures (only those that are over 4). The 3/4 and 4/4 signatures seem to work well on our database, but relying on this library for more "exotic" musical content is difficult (for example, Irish traditional music contains a lot of *jig* in 6/8 or *slides* in 12/8).

## 5. NOISE ROBUSTNESS

Algorithms in digital communications are usually rated through their robustness to noise. Likewise here, the idea is to evaluate the efficiency of the various inference methods on "noise-corrupted" chroma. The flow-chart of the corresponding system is represented in Figure 5. We work with

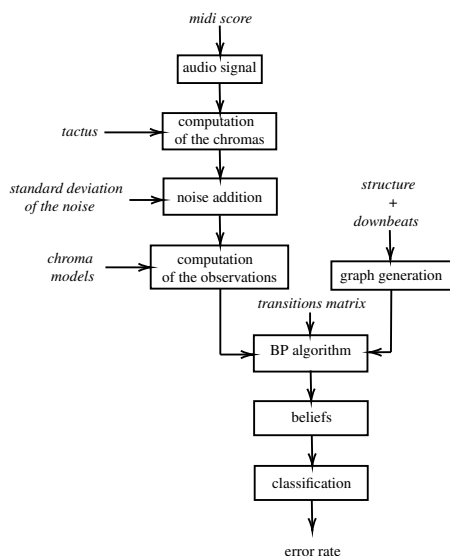


Figure 5. Flowchart of the test system for estimating the robustness to noise.

a simple midi-track made up of 8 verses containing 4 bars in 4/4 of Em, C, G and D, repeated 4 times each (one distinct chord per bar), resulting in a total of 128 chords. The midi partition is then converted at 60 BPM to CD quality audio using the *grand piano* virtual instrument of Ableton Live. Chroma are then extracted using the Python Library Librosa [26]. First the harmonic part is extracted with the function *harmonic(y=y, margin=5)* (see [27]) and then CQT-chromas are computed with the function *feature.chroma\_cqt*. Finally, an average chroma for each beat is computed.

Gaussian Noise with a standard deviation of  $\sigma$  is then added to the chroma vectors and the observations vectors are computed as in [28] from the corrupted chroma. For each algorithm and each  $\sigma$ , a set of 100 corrupted chroma vectors is generated and the average chord estimation error rate is recorded. The results are presented in Figure 6. We see that the max-product BP clearly beats the sum-product flavour.

All in all, we argue that adding noise to the chroma allows for blending the whole complexity of music into the performance estimation process: chroma vectors are indeed sensitive to arrangements, e.g., percussive events that may

randomize the distribution of chroma components. In addition, we have shown that adding long-term constraints improves the overall robustness of the inference process over such perturbations.

## 6. RESULTS AND DISCUSSION

Chroma and observation probabilities are computed with Matlab while subsequent steps are implemented in Julia [29].

The results over real world datasets are presented in Table 1. Observations are the same for each row. HMM refers to the Viterbi algorithm (see 2.2), while BP refers to the Belief Propagation decoder using the perceptual matrix. Rows 3, 4 and 5 take downbeats or structure information or both into account (see Figure 4), respectively. Row 6 also includes the "cycle of fifths" transition matrix, while row 7 includes similarities as explained in Section 3.3.

Results with the perceptual transitions matrix and those with the cycle of fifth transitions matrix are very close. An unpaired t-test gives a probability  $p=0.98$  for the null hypothesis at 95%: the groups are not statistically different. In addition, the same occurs for "BP both" and "BP both (correlation)" ( $p=0.54229$ ).

System	Ground truth	Estimated
HMM (Viterbi)	71.31 %	70.45 %
BP (perceptual)	71.36%	70.03 %
BP with downbeats	73.76%	71.9%
BP with structure	72.53%	-
BP both	75.32%	73.65%
BP both (cycle of fifths)	75.35%	-
BP both (correlation)	75.09%	-
State of the Art	86.80 %	

Table 1. Performance of the various algorithms using ground truth beats and downbeats. The *ground truth* column shows the results obtained with ground truth beats, downbeats and structures whereas the *Estimated* column shows those obtained with estimated information. The state of the art system is the system achieving the best performances for MIREX 2018 on the Isophonics dataset with the Maj/min dictionary (FK2 system, by Florian Krebs, Filip Korzeniowski, Sebastian Bck).

These interesting results have to be nuanced however. While in some songs the recognition rate may reach 95%, other yield result lower than 50%. Two main reasons have been identified. First, the restriction of the dictionary (non major/minor chords that have not been well mapped to the major-minor equivalent) leads to computational errors but the chords proposed by the BP decoder are musically acceptable. Second, instead of identifying each chord on all its duration, the algorithm tends to oscillate between two states that are related to the ground truth chord. The crucial importance of the conditional probabilities between states is exemplified here: whenever the self-transition probability is increased, the previous issue vanishes but then short chord transitions will not be detected. On the contrary, whenever the self transition is lowered, short chord transitions are well detected but long time chords undergo poor detection.

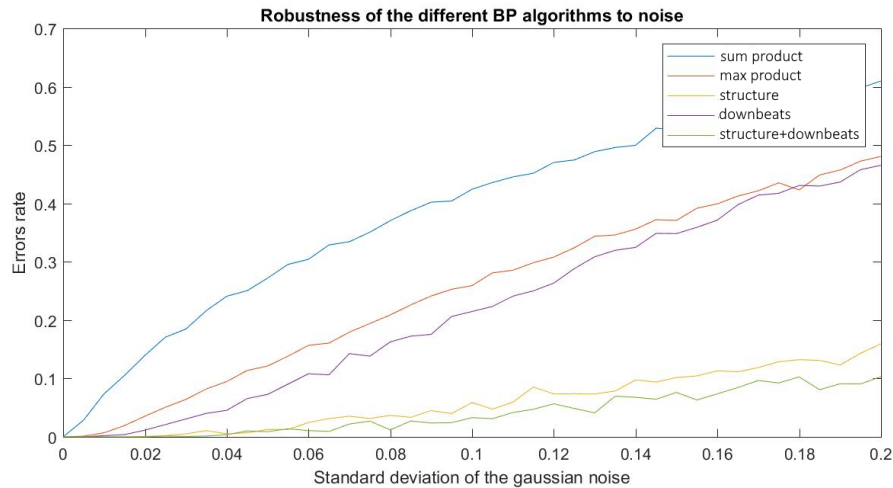


Figure 6. Chord estimation error rate of the various algorithms vs noise amplitude.

It should also be noted that with estimated beats and downbeats, results are worse than those using ground truth but BP still stays ahead.

## 7. METHODOLOGY DISCUSSION

The measure of the performance of a system is usually done by the recall of the ground truth. But the ground truth itself depends on the people that elaborate it. In [30], it was attempted to elaborate a system that would take the subjectivity of the annotators into account. This practice should be given more attention in the following years. In [3] the authors attempted to compare the results of their systems with ground truth but also with two independent annotators. Their results show that WCSR is not the best criterion to measure the performance of a system and that above a certain threshold, an apparently acceptable WCSR does not make sense any more if it is larger than that of the annotator.

In addition, as pointed out by [31], the fact that the dictionary is limited to the 24 major and minor chords can cause some further errors. The aim of this work is not to enhance the performance of the whole system but only the pattern matching part: HMM and the various BP algorithms are compared against the same dictionary and the same features. Only a few systems use large dictionaries.

## 8. CONCLUSION AND FUTURE WORK

We proposed a new approach to the inference step in ACE that outperforms the HMM one. While the current trend is to develop deep-learning based system, our method does not require training, hence does not imply large annotated databases. The architecture of our system only uses structure information and downbeats from the ground truth.

Although we do not make use of this feature in the present work, it is worth mentioning that the very general formulation of the BP algorithm makes it possible to feed any N-point correlation function into the iteration process, i.e., one that would describe higher-level correlations between

tuples of chords, not just pairs. This opens up the way to taking complex musical context into account.

Future works about this project may include studying the influence of the graph girth on the stability of the iteration process and on the computation time. Relying on the Generalized Belief Propagation algorithm [32] might be a way to improve the robustness of the system by suppressing oscillating behaviors. Finally, it would be promising to investigate further how using similarities between observed chroma could help improving the efficiency of the algorithm.

## 9. ACKNOWLEDGMENTS

This material is based upon work supported by Laboratoire ETIS, UMR 8051, CNRS/ENSEA/Université de Cergy-Pontoise and Université Paris-Seine.

## 10. REFERENCES

- [1] M. McVicar, R. Santos-Rodriguez, Y. Ni, and T. De Bie, "Automatic Chord Estimation from Audio: A Review of the State of the Art," in *IEEE/ACM TASLP*, 2014.
- [2] B. L. Sturm, "Revisiting Priorities: Improving MIR Evaluation Practices," in *17th ISMIR*, 2016.
- [3] E. J. Humphrey, J. P. Bello, and T. Cho, "Four timely insights on automatic chord estimation," in *16th ISMIR*, 2015.
- [4] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, "Automatic music transcription: Challenges and future directions," *Journal of Intelligent Information Systems*, 2012.
- [5] M. Mauch, "Automatic Chord Transcription from Audio Using Computational Models of Musical Context," Ph.D. dissertation, Queen Mary University of London, 2010.
- [6] H. Papadopoulos and G. Peeters, "Joint Estimation of chords and downbeats from an audio signal," in *IEEE - TASLP*, 2009.
- [7] R. Christopher and J. Stoddard, "Harmonic analysis with probabilistic graphical models," in *4th ISMIR*, 2003.

- [8] H. Papadopoulos and G. Tzanetakis, "Models for Music Analysis From a Markov Logic Networks Perspective," in *IEEE/ACM TASLP*, Jan. 2017.
- [9] S. Sigtia, E. Benetos, N. Boulanger-Lewandowski, T. Weyde, A. S. d'Avilar Garcez, and S. Dixon, "A Hybrid Recurrent Neural Network For Music Transcription," in *IEEE - ICASSP*, 2015.
- [10] F. Korzeniowski and G. Widmer, "On the Futility of Learning Complex Frame-Level Language Models for Chord Recognition," in *AES Conf. on Semantic Audio*, 2017.
- [11] J.-F. Paiment, D. Eck, and S. Bengio, "A Probabilistic Model for Chord Progressions," in *6th ISMIR*, 2005.
- [12] M. Mezard and A. Montanari, *Information, Physics and Computation*, 2009.
- [13] D. Lang and N. de Freitas, "Beat tracking the graphical model way," in *NIPS*, 2004.
- [14] T. Cho and J. P. Bello, "On the Relative Importance of Individual Components of Chord Recognition Systems," in *IEEE/ACM TASLP*, 2014.
- [15] J. A. Bilmes, "Techniques to foster drum machine expressivity," in *Int. Comp. Music Conf.*, 1993.
- [16] H. Papadopoulos, "Joint Estimation Of Musical Content Information From An Audio Signal," Ph.D. dissertation, 2010.
- [17] C. L. Krumhansl, *Cognitive Foundations of Musical Pitch*, 1990.
- [18] Y. W. Jonathan S. Yedidia, William T. Freeman, *Exploring artificial intelligence in the new millennium*, 2003, ch. Understanding belief propagation and its generalizations, pp. 239–269.
- [19] J. Coughlan, "A Tutorial Introduction to Belief Propagation," 2009.
- [20] H. Papadopoulos and G. Peeters, "Local Key Estimation from an Audio Signal Relying on Harmonic and Metrical Structures," in *IEEE - TASLP*, 2011.
- [21] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *IEEE Int. Conf. on Multimedia and Expo*, 2000.
- [22] B. L. Sturm, R. Bardeli, T. Langlois, and V. Emiya, "Formalizing The Problem Of Music Description," in *15th ISMIR*, 2014.
- [23] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "mir\_eval: a transparent implementation of common MIR metrics," in *15th ISMIR*, 2014.
- [24] J. Pauwels and G. Peeters, "Evaluating automatically estimated chord sequences," in *IEEE - ICASSP*, 2013.
- [25] S. Bock, F. Korzeniowski, J. Schlter, F. Krebs, and G. Widmer, "madmom: a new Python Audio and Music Signal Processing Library," in *24th ACM Int. Conf. on Multimedia*, 2016.
- [26] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and Music Signal Analysis in Python," in *SCIPY*, 2015.
- [27] J. Driedger, M. Muller, and S. Dish, "Extending Harmonic-Percussive Separation of Audio Signals," in *15th ISMIR*, 2014.
- [28] E. Gomez, "Tonal Description of Polyphonic Audio from Music Content Processing," *INFORMS Journal on Computing*, 2006.
- [29] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah, "Julia: A Fresh Approach to Numerical Computing," *SIAM Reviews*, pp. 65–98, 2017.
- [30] H. V. Koops, W. de Haas, J. Bransen, and A. Volk, "Chord Label Personalization through Deep Learning of Integrated Harmonic Interval-based Representations," in *Proc. of the First Int. Workshop on Deep Learning and Music joint with IJCNN*, May 2017.
- [31] E. J. Humphrey, T. Cho, and J. P. Bello, "Learning a robust tonnetz-space transform for automatic chord recognition," in *IEEE - ICASSP*, 2012.
- [32] S. Reynal, J.-C. Sibel, and D. Declercq, "An Application of Generalized Belief Propagation: Splitting Trapping Sets in LDPC Codes," in *IEEE Int. Symposium on Information Theory*, 2014.

# HMM-BASED GLISSANDO DETECTION FOR RECORDINGS OF CHINESE BAMBOO FLUTE

Changhong Wang<sup>1</sup>, Emmanouil Benetos<sup>1</sup>, Xiaojie Meng<sup>2</sup>, Elaine Chew<sup>1</sup>

<sup>1</sup>Centre for Digital Music, Queen Mary University of London, UK

{changhong.wang, emmanouil.benetos, elaine.chew}@qmul.ac.uk

<sup>2</sup>Department of Chinese Music, China Conservatory of Music, China

mxj.yd@foxmail.com

## ABSTRACT

Playing techniques such as ornamentations and articulation effects constitute important aspects of music performance. However, their computational analysis is still at an early stage due to a lack of instrument diversity, established methodologies and informative data. Focusing on the Chinese bamboo flute, we introduce a two-stage glissando detection system based on hidden Markov models (HMMs) with Gaussian mixtures. A rule-based segmentation process extracts glissando candidates that are consecutive note changes in the same direction. Glissandi are then identified by two HMMs. The study uses a newly created dataset of Chinese bamboo flute recordings, including both isolated glissandi and real-world pieces. The results, based on both frame- and segment-based evaluation for ascending and descending glissandi respectively, confirm the feasibility of the proposed method for glissando detection. Better detection performance of ascending glissandi over descending ones is obtained due to their more regular patterns. Inaccurate pitch estimation forms a main obstacle for successful fully-automated glissando detection. The dataset and method can be used for performance analysis.

## 1. INTRODUCTION

Computational analysis of expressive patterns in music signals plays an important role in music information research. For instrumental music, these expressive patterns are frequently the result of playing techniques. Automated analysis of playing techniques can benefit automatic music transcription [1], computer-aided music pedagogy [2], instrument classification [3, 4], and performance analysis [5]. However, computational analysis of playing techniques is still in its early stages, lacking instrument diversity, established methodologies, and informative data.

Most existing work on computational analysis of playing techniques focuses on Western instruments such as guitar [6–8], violin [9–11], piano [12], and drums [13, 14].

C. Wang is funded by the China Scholarship Council (CSC). E. Benetos is supported by a UK RAEng Research Fellowship (RF/128).

Copyright: © 2019 Changhong Wang, Emmanouil Benetos, Xiaojie Meng, Elaine Chew. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Playing techniques in non-Western instruments, while similarly important, are often overlooked. Take for example, one of the world's most ancient instruments, the Chinese bamboo flute (also known as the *Dizi* or *Zhudi*, thereafter referred to as CBF): many listeners are most often captivated by its unique timbre, which belies the twenty or more playing techniques invoked when performing on the instrument. To our knowledge, only Ayers [15, 16] has done some analysis of CBF playing techniques through synthesis. This work focused only on trills, tremolos and flutter-tongue. But many other techniques remain to be explored. For the case of other non-Western instruments, limited computational work can be found [5, 17].

For playing technique detection, methods adopted in the literature are typically frame-wise classifiers based on high dimensional feature inputs [6, 18], with little explanation of why the methods work. Support vector machines (SVMs) are the most frequently used class of methods. A series of electric bass guitar playing techniques was classified into plucking or expressive styles using SVMs in [6]; [10] applied it to distinguish five fundamental guitar playing techniques. A multimodal input using SVMs was used for analysing piano pedalling techniques in [12]. Su et al. [11] proposed new features as input to an SVM based on sparse modeling of magnitude and phase-derived spectra before classifying violin playing techniques. Other work used dynamic time warping [19], COSFIRE filters [20], spectrogram templates [21], and filter diagonalisation method [22] for analysis of playing techniques.

Datasets used in playing technique research consist of mainly playing techniques performed in isolation. Isolated techniques can vary greatly from the same techniques used in live performance. For ecological validity, we argue that playing techniques should be collected in context. A challenge of obtaining playing technique examples in real-world settings is that some techniques may be rare. Thus, it may be hard to find pieces covering a wide range of playing techniques and with sufficient repeated instances of these techniques to obtain a variety of samples for a specific technique.

To address these limitations, we use the CBF as our instrument of choice and glissando, a rarely explored audio gesture in the literature, as our starting point, aiming to build a systematic methodology for automatically analysing playing techniques. *Glissando*, here refers to a rapid slide up or down the musical scale [23], which is not



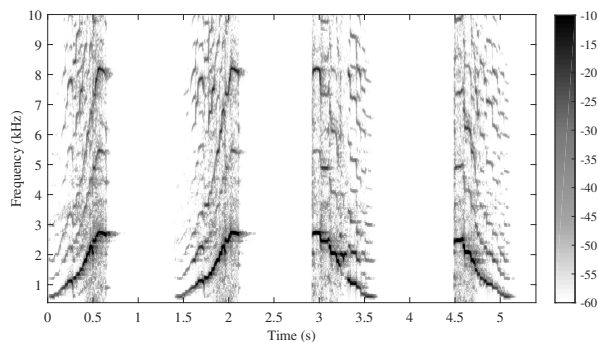


Figure 1. Spectrogram of two ascending and two descending glissando examples in Chinese bamboo flute music.

comparable to the one defined as a continuous slide from one note to another in [24]. Fig. 1 shows a spectrogram of a series of two ascending and two descending CBF glissandi. As can be seen, they exhibit a readily recognisable pattern, resembling rapid scale segments. Glissando detection in CBF playing is not straightforward: CBF glissandi are less regular than the stair-like glissando patterns in piano and guitar playing [18]. For the same glissando type, variations exist in the ways they are executed between different players, different pieces, and even different parts of the same piece. The main characteristic of glissando is the consecutive note change, which we claim can be captured by latent states of a hidden Markov model (HMM) [25, 26]. HMMs enable the decoding of note evolution while smoothing outlier variations within performed glissandi.

In this paper, we make a first attempt to the computational analysis of CBF glissandi. A new dataset including both isolated glissandi and real-world pieces is created and is being prepared for public release. Based on the analysis of ground truth statistics, we propose a two-stage detection system. A rule-based segmentation process first extracts glissando candidates that are consecutive note changes in the same direction. Different from traditional binary classification, the false positives obtained in the segmentation stage, which exhibit similar pitch evolution and duration as the ground truth, are used to train a non-glissando HMM (NG-HMM). A glissando HMM (G-HMM) is trained using all ground truth glissandi in the training set. Glissandi are then identified by two HMMs at test time.

## 2. DATASET

### 2.1 Dataset Information

The glissando analysis dataset, CBF-GlissDB, comprises recordings by ten expert CBF players from the China Conservatory of Music. All data is recorded in a professional recording studio using a Zoom H6 recorder at 44.1kHz/24-bits. Each player performs both isolated glissandi covering all notes on the CBF and one full-length piece—*Busy Delivering Harvest* 《扬鞭催马运粮忙》 or *Morning* 《早晨》. Players are grouped by flute type (C and G, the most representative types for Southern and Northern styles, respectively) and each player uses their own flute. Details of

recording length and number of glissandi in each group are shown in Table. 1.

Players	Flute	Isolated glissandi		Whole-piece recordings	
		Length (mins)	#glissandi [↑, ↓]	Piece, style	Length #glissandi (mins) [↑, ↓]
1-3	C	2.4	[58,47]	<i>Morning</i> , Southern	16.0 [24,0]
4-10	G	5.0	[117,112]	<i>Busy Delivering Harvest</i> , Northern	28.0 [23,106]

Table 1. Dataset information.

In order to assess the performance of the proposed glissando detection system independent of the performance of pitch estimation methods, pitch ground truth for all recordings is created. The fundamental frequency of each recording is first estimated using the pYIN algorithm [27] due to the strictly monophonic property of the recordings. All errors are then manually corrected by the first author using Sonic Visualiser<sup>1</sup>. Both isolated and performed glissandi are annotated and verified by the players on the score. The final annotation is created by the first author after consulting with the players.

### 2.2 Dataset Statistics

To verify the intuition of the difference between isolated and performed glissandi, characteristic statistics of the ground truth are calculated. Fig. 2 shows two-dimensional histograms for four types of glissandi in CBF-GlissDB: ascending and descending isolated glissandi; and ascending and descending performed glissandi. As can be seen, performed glissandi have shorter durations than isolated glissandi, especially for descending glissandi, performed ones have almost half duration as isolated ones. Further analysis of note durations within each glissandi shows little difference among isolated glissandi while ascending performed glissandi have larger variation than descending performed ones. This may be attributed to the performers' tendency to lengthen the start or end note in an ascending performed glissando.

## 3. METHOD

To automatically detect glissando from real-world CBF recordings, we propose a two-stage detection system based on rule-based segmentation (Sec. 3.1) and HMM-based identification (Sec. 3.2).

### 3.1 Rule-based Segmentation

To obtain glissando candidates from the whole-piece recordings, we introduce a rule-based segmentation component using pitch with a 20ms hop size as input, as demonstrated in Fig. 3. The pitch is first smoothed to exclude noisy variations and quantised to the nearest notes in 12-tone equal temperament scale, resulting in 16 notes in the CBF tonal

<sup>1</sup> <https://www.sonicvisualiser.org>

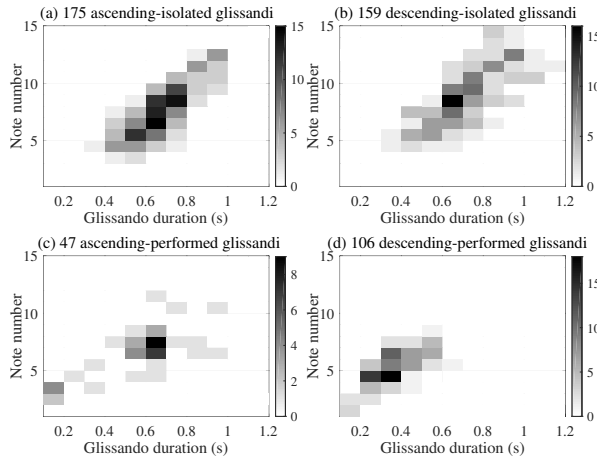


Figure 2. Duration and note number histograms for four glissando types.

range: G4-A6 for the C flute, and D5-E7 for the G flute (we assume that flute types are known for the current system). Frames with pitch less than 250Hz and waveform amplitude less than -20dB are marked as silence. The sign of note change is extracted to represent note change direction. Consecutive note changes in the same direction are then extracted as glissando candidates, which are further pruned by constraints on note numbers (at least 4 for both ascending and descending glissandi) and duration (at least 0.2s for ascending glissandi and 0.15s for descending glissandi based on the consultations with the professional players).

### 3.2 HMM-based Identification

#### 3.2.1 Feature Extraction

Since all glissando candidates (extracted in the previous stage) share similar pitch evolution characteristics, the input to the HMMs must possess sufficient discriminative power to distinguish glissandi from non-glissandi. Considering the pitch discreteness and long duration of glissandi, we use a feature set consisting of both short-term (average pitch change, average intensity, average intensity change) and long-term (note number, note duration, note range) features [28, 29]. All features are statistics (mean and standard deviation) of pitch and intensity with variations on window and hop sizes. Hop size variations range from 10ms to 20ms at intervals of 2ms, while window sizes depend on the glissando direction.

##### (i) Short-term features:

To capture pitch and intensity change, the short-term window varies from 100 to 200ms at intervals of 20ms for the following three features.

##### – Average pitch change:

$$\Delta p_i = \frac{1}{w} \sum_{k=1}^w [p_i(k) - p_{i-1}(k)], \quad (1)$$

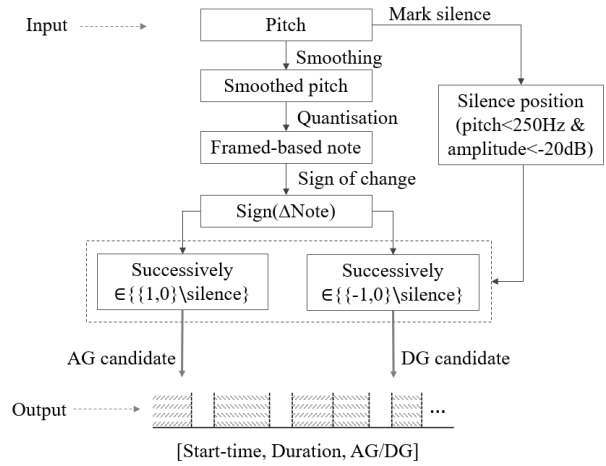


Figure 3. Diagram of rule-based segmentation (AG=ascending glissando; DG=descending glissando).

where  $p_i(k)$  is the  $k$ -th pitch value within the window centered at the  $i$ -th time frame, and  $w$  is the window length.

##### – Average intensity (amplitude in dB scale) [7]:

$$I_i = \frac{1}{w} \sum_{k=1}^w \left[ 20 \cdot \log_{10} A_i(k) \right], \quad (2)$$

where  $A_i(k)$  is the amplitude of the  $k$ -th sample within the window centered at the  $i$ -th time frame, and  $I_i$  is average intensity of this window.

##### – Average intensity change: $\Delta I_i = I_i - I_{i-1}$ .

##### (ii) Long-term features:

To capture the discreteness of pitch evolution, note-level features with long windows are calculated. The window sizes vary from 200 to 400ms at intervals of 50ms for descending glissandi with shorter duration, and from 200 to 600ms at the same intervals for ascending glissandi which have longer duration. The calculation process for one ascending glissando example is shown in Fig. 4. With a 400ms window sliding forward, the number of notes  $N$  is 8 (one more than the number of peaks, highlighted by the red circles) and note range (note change between start and end notes)  $R$  equals 7. Note durations  $D$ , which refer to the intervals between two note change peaks, are  $\{80, 40, 60, 40, 40, 60\}$ ms.

#### 3.2.2 HMM-based Identification

As shown in Fig. 5, two HMMs with Gaussian mixture emissions are trained on the training set, with k-means initialisation and iterative parametrisation by the Expectation-Maximization algorithm [30]. During the training process, model parameters—the number of HMM latent states, number of Gaussian mixture components, and window-hop sizes—are varied and the model with the best performance on

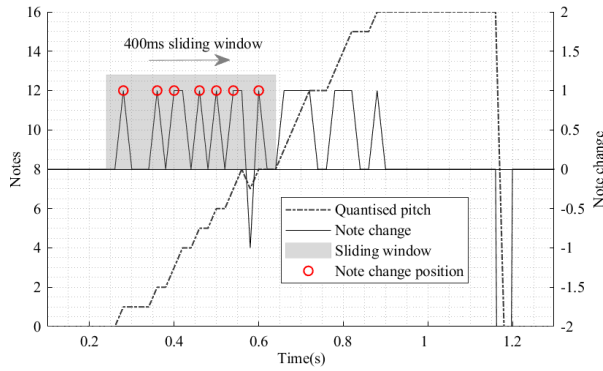


Figure 4. Long-term feature calculation process based on one example of an ascending glissando.

the validation set is chosen as the final one for testing. The emission used is a Gaussian mixture distribution [30]:

$$p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{m=1}^M \pi_m \mathcal{N}(\mathbf{x}_i|\mu_m, \Sigma_m), \quad (3)$$

where  $\mathbf{x}_i$  is the observed feature vector of the  $i$ -th frame;  $\pi_m$ ,  $\mu_m$  and  $\Sigma_m$  are the prior, mean and covariance of the  $m$ -th mixture component; and  $\boldsymbol{\pi}$ ,  $\boldsymbol{\mu}$ ,  $\boldsymbol{\Sigma}$  are the model parameters, each of which is an  $M$ -dimensional vector corresponding to  $\pi_m$ ,  $\mu_m$ , and  $\Sigma_m$ .

The CBF-GlissDB is subdivided into three subsets, namely, training (all isolated glissandi and 6 whole pieces), validation (2 whole pieces), and test (2 whole pieces). The segmentation stage is applied to whole-piece recordings in all three subsets, but to different ends. For the training set, segmentation serves the purpose of obtaining false positives that are then used to train a NG-HMM. In the validation and test stages, the extracted segments serve as candidates to be assigned glissando (G) or non-glissando (NG) labels by comparing the log-likelihood calculated by the two HMMs. Since the HMMs are applied directly to the candidate segments, the absolute position of glissandi in the pieces does not influence the result. The ten whole-piece recordings are randomly allocated to the training, validation, and test sets in a 6:2:2 ratio at the beginning of experiment. A five-fold cross-validation is then conducted.

#### 4. EVALUATION

To investigate the influence of automatic pitch estimation on glissando detection, evaluation of both a semi-automated system (using the pitch ground truth as input) and a fully-automated system (using pitch automatically estimated by pYIN [27] as input) is carried out. Because glissando length ranges approximately from 200 to 1100ms, for each system, frame-based and segment-based evaluations are implemented. The frame size used in frame-based evaluation is 20ms. Segment-based evaluation compares detected glissandi and ground truth in short-time, non-overlapping segments [31]. A segment length of 100ms is adopted. True positives are segments which have overlaps with both

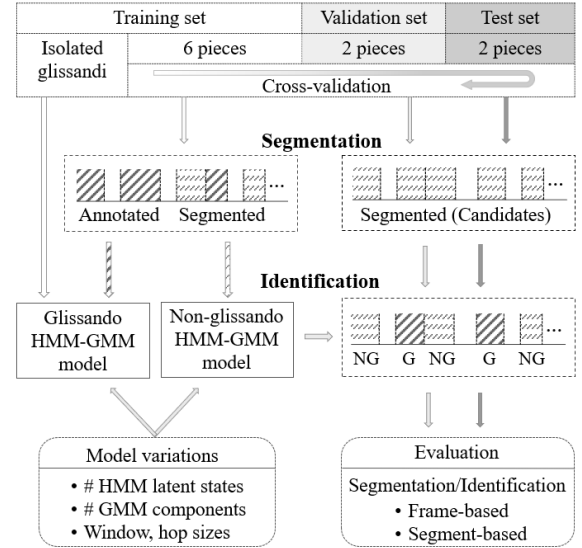


Figure 5. System diagram for glissando detection (G=glissando; NG=non-glissando).

ground truth and detected glissandi; false positives segments overlaps only with detected glissandi; and, false negatives intersect with ground truth only.

#### 4.1 Semi-automated System Evaluation

Table 2 gives the precision, recall, and F-measure results for both ascending and descending glissandi in the semi-automated detection system. As can be seen, the segmentation stage performs a conservative selection of candidate segments with high recall and low precision. The large number of false positives obtained for NG-HMM training benefits the data balance in our system. The better identification performance of ascending glissandi over descending ones can be attributed to their more regular patterns. As can be seen, the identification F-measure increased by approximately 60% as compared to the segmentation F-measure, which verifies our intuition that consecutive pitch evolution can be captured by HMMs.

Stage	Glissando direction	Frame-based (%)			Segment-based (%)		
		$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Rule-based segmentation	Ascending	3.1	93.4	5.9	3.1	92.8	6.0
	Descending	4.9	83.1	9.0	5.1	86.9	9.9
HMM-based Identification	Ascending	73.4	75.4	73.4	72.0	74.0	72.0
	Descending	65.4	67.6	63.2	64.4	70.2	64.2

Table 2. Evaluation results of the semi-automated glissando detection system based on annotated pitch ( $\mathcal{P}$ =precision,  $\mathcal{R}$ =recall,  $\mathcal{F}$ =F-measure).

#### 4.2 Fully-automated System Evaluation

After verifying the proposed glissando detection method independently, we then use the automatically estimated pit-

ch to evaluate the fully-automated glissando detection system. Due to the influence of breathing, some parts in the CBF recordings have high intensity but no detected pitch. Thus silence cannot be determined only by pitch presence, and we define silence bits as parts having both no pitch and intensity below -20dB. Correctly detected frames are the voiced parts with pitch intervals less than half a semitone between the ground truth and the detected pitch. Pitch estimation accuracy refers to the percentage of correctly detected frames over all voiced frames. Table 3 shows the estimated pitch result of both whole-piece recordings and ground truth glissando segments within these pieces. The poorer pitch estimation performance on glissando segments shows that pYIN works less well on rapid pitch evolution progressions.

Type	Whole pieces		Glissando segments	
	Southern	Northern	Ascending	Descending
Accuracy (%)	80.2	79.5	72.0	74.8

Table 3. Pitch estimation accuracy for whole-piece recordings and glissando segments.

The fully-automated glissando detection results are shown in Table 4. Considering the pitch evaluation shown above, it is reasonable to expect worse performance when using automatically estimated pitch as input. Pitch is a main discriminative feature in the proposed glissando detection system. The presence of undetected pitches or octave errors within glissandi hinders G-HMM to capture the consecutive note evolution. Thus false positives, which exhibit similar pitch evaluation as the ground truth glissandi and have higher pitch estimation accuracy, may be assigned with G labels. This is verified by the better identification performance on descending glissandi over ascending ones with lower pitch estimation result.

Stage	Glissando direction	Frame-based (%)			Segment-based (%)		
		$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
Rule-based segmentation	Ascending	2.1	84.8	4.1	2.1	86.2	4.4
	Descending	3.3	67.3	5.9	3.6	75.0	7.1
HMM-based identification	Ascending	36.4	63.2	44.6	36.8	63.4	45.0
	Descending	58.2	48.4	50.4	58.0	51.8	52.6

Table 4. Evaluation results of the fully-automated glissando detection system based on estimated pitch.

## 5. CONCLUSIONS

In this paper, we have described a first attempt at computational analysis of CBF glissandi. HMMs are introduced to decode the consecutive note evolution within glissandi and a two-stage detection system is proposed. Using inputs based only on the statistics of two low-level features—pitch and intensity, frame- and segment-based F-measures of 73.4% and 72.0% for ascending glissandi, and 63.2%

and 64.2% for descending glissandi, are obtained in a semi-automated detection system, which confirms the feasibility of our method for glissando detection. The poorer performance of the fully-automated system may be attributed to the inaccuracy of pitch estimation since pitch is the main discriminative feature.

Future work will seek to implement other state-of-art pitch estimation methods (for example, CREPE [32]) to improve pitch detection accuracy prior to glissando detection. More informative features for glissando description may be explored. Alternative methods for glissando identification will be investigated, such as template-based detection, the spiral scattering transform [33], and deep learning, the latter including data augmentation of the collected audio samples. Plans are underway for expansion of the dataset. The analysis will also be expanded to other CBF playing techniques, with the aim to develop a systematic methodology for CBF playing technique detection.

## 6. REFERENCES

- [1] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, “Automatic music transcription: challenges and future directions,” *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 407–434, 2013.
- [2] Y. Han and K. Lee, “Hierarchical approach to detect common mistakes of beginner flute players,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 77–82.
- [3] G. E. Hall, H. Ezzaidi, M. Bahoura, and C. Volat, “Classification of pizzicato and sustained articulations,” in *European Signal Processing Conference (EUSIPCO)*, 2013, pp. 1–5.
- [4] V. Lostanlen, J. Andén, and M. Lagrange, “Extended playing techniques: the next milestone in musical instrument recognition,” in *5th International Conference on Digital Libraries for Musicology*, 2018.
- [5] L. Yang, “Computational modelling and analysis of vibrato and portamento in expressive music performance,” Ph.D. dissertation, Queen Mary University of London, 2017.
- [6] J. Abeßer, H. Lukashevich, and G. Schuller, “Feature-based extraction of plucking and expression styles of the electric bass guitar,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010, pp. 2290–2293.
- [7] J. Abeßer and G. Schuller, “Instrument-centered music transcription of solo bass guitar recordings,” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 25, no. 9, pp. 1741–1750, 2017.
- [8] L. Su, L. F. Yu, and Y. H. Yang, “Sparse cepstral and phase cdes for guitar playing technique classification,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 9–14.



- [9] I. Barbancho, C. Bandera, A. M. Barbancho, and L. J. Tardon, "Transcription and expressiveness detection system for violin music," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 189–192.
- [10] S. H. Chen, S. H. Wu, Y. S. Lee, R. Lo, and J. C. Wang, "Hierarchical representation based on Bayesian non-parametric tree-structured mixture model for playing technique classification," in *Proceedings of the Thematic Workshops of ACM Multimedia*, 2017, pp. 537–543.
- [11] L. Su, H. M. Lin, and Y. H. Yang, "Sparse modeling of magnitude and phase-derived spectra for playing technique classification," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 22, no. 12, pp. 2122–2132, 2014.
- [12] B. Liang, G. Fazekas, A. McPherson, and M. Sandler, "Piano pedaller: a measurement system for classification and visualisation of piano pedalling techniques," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, 2017, pp. 325–329.
- [13] P. Herrera, A. Yeterian, and F. Gouyon, "Automatic classification of drum sounds: a comparison of feature selection methods and classification techniques," in *International Conference on Music and Artificial Intelligence (ICMAI)*, 2002, pp. 69–80.
- [14] M. Prockup, E. M. Schmidt, J. J. Scott, and Y. E. Kim, "Toward understanding expressive percussion through content based analysis," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2013, pp. 143–148.
- [15] L. Ayers, "Synthesizing trills for the Chinese dizi," in *International Computer Music Conference (ICMC)*. Singapore, 2003, pp. 227–30.
- [16] —, "Synthesizing timbre tremolos and flutter tonguing on wind instruments," in *International Computer Music Conference (ICMC)*. Miami, Florida, USA, 2004.
- [17] T. H. Özasan, X. Serra, and J. L. Arcos, "Characterization of embellishments in new performances of makam music in Turkey," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2012, pp. 13–18.
- [18] Y. P. Chen, L. Su, and Y. H. Yang, "Electric guitar playing technique detection in real-world recording based on F0 sequence pattern recognition," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2015, pp. 708–714.
- [19] S. Giraldo and R. Ramírez, "Performance to score sequence matching for automatic ornament detection in jazz music," in *International Conference of New Music Concepts (ICMNC)*, 2015.
- [20] A. Neocleous, G. Azzopardi, C. N. Schizas, and N. Petkov, "Filter-based approach for ornamentation detection and recognition in singing folk music," in *International Conference on Computer Analysis of Images and Patterns (CAIP)*, 2015, pp. 558–569.
- [21] J. Driedger, S. Balke, S. Ewert, and M. Müller, "Template-based vibrato analysis in music signals," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 239–245.
- [22] L. Yang, K. Rajab, and E. Chew, "The filter diagonalisation method for music signal analysis: frame-wise vibrato detection and estimation," *Journal of Mathematics and Music*, vol. 11, no. 1, pp. 42–60, 2017.
- [23] Merriam-Webster, *Webster's ninth new collegiate dictionary*. Merriam-Webster, 1983.
- [24] R. Panda, R. Malheiro, and R. P. Paiva, "Novel audio features for music emotion recognition," *IEEE Transactions on Affective Computing*, no. 1, pp. 1–1, 2018.
- [25] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
- [26] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 1, pp. 40–48, 2010.
- [27] M. Mauch and S. Dixon, "pYIN: A fundamental frequency estimator using probabilistic threshold distributions," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 659–663.
- [28] P. C. Li, Y. H. Y. L. Su, and A. W. Y. Su, "Analysis of expressive musical terms in violin using score-informed and expression-based audio features," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2015, pp. 809–815.
- [29] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project." technical report, IRCAM, Paris, France, Apr. 2004.
- [30] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT Press, 2013.
- [31] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.
- [32] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "Crepe: A convolutional representation for pitch estimation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 161–165.
- [33] V. Lostanlen and S. Mallat, "Wavelet scattering on the pitch spiral," in *18th International Conference on Digital Audio Effects (DAFx)*, 2015.

# Towards CNN-based Acoustic Modeling of Seventh Chords for Automatic Chord Recognition

Christon-Ragavan Nadar<sup>1</sup>, Jakob Abeßer<sup>1</sup>, Sascha Grollmisch<sup>1,2</sup>

<sup>1</sup>Semantic Music Technologies Group, Fraunhofer IDMT, Ilmenau, Germany

<sup>2</sup>Institute of Media Technology, Technische Universität Ilmenau, Ilmenau, Germany  
jakob.abesser@idmt.fraunhofer.de

## ABSTRACT

In this paper, we build upon a recently proposed deep convolutional neural network architecture for automatic chord recognition (ACR). We focus on extending the commonly used major/minor vocabulary (24 classes) to an extended chord vocabulary of seven chord types with a total of 84 classes. In our experiments, we compare joint and separate classification of the chord type and chord root pitch class using one or two separate models, respectively. We perform a large-scale evaluation using various combinations of training and test sets of different timbre complexity. Our results show that ACR with an extended chord vocabulary achieves high f-scores of 0.97 for isolated chord recordings and 0.66 for mixed contemporary popular music recordings. While the joint ACR modeling leads to the best results for isolated instrument recordings, the separate modeling strategy performs best for complex music recordings. Alongside with this paper, we publish a novel dataset for extended-vocabulary chord recognition which consists of synthetically generated isolated recordings of various musical instruments.

## 1. INTRODUCTION

Automatic chord recognition (ACR) has been actively researched in the field of Music Information Retrieval (MIR) during the last 20 years. ACR algorithms are an essential part of many music applications such as music transcription systems for automatic lead-sheet generation, music education and learning applications, as well as music similarity and recommendation algorithms. In music practice, chord sequences can be played as different chord voicings (selection and order of chord tones) on a large variety of musical instruments, each with its own unique sound characteristic. Therefore, the biggest challenge in ACR is to extract the predominant harmonic changes in a music signal while being robust against different instrument timbres. Furthermore, tuning deviations of music recordings as well as inherent ambiguities between different chords can complicate the task even more [1]. In general, ACR is

approached as a two-step problem. First, the acoustic modeling step deals with the prediction of chord labels from short-term audio signal frames. Secondly, during the temporal modeling step, post-processing algorithms are applied to merge frame-level predictions to longer segment-level chord annotations.

As the first main contribution of this paper, we investigate the under-explored task of recognizing seventh chords as an extension to commonly used major and minor chords. Most previous publications focus on recognizing the 24 possible major and minor chords. In the extended-vocabulary ACR scenario, we investigate 7 different chord types including four seventh chord types and the power-chord, which leads to a total of 84 classes. Throughout this paper, we solely focus on improving the acoustic modeling for ACR and do not apply any temporal modeling algorithms. As a second contribution, we compare joint and separate modeling of the chord root pitch class and the chord type as two possible strategies for ACR which are described in Section 3.2. Finally, we publish a novel dataset alongside with this paper that includes synthetically generated chord sequences of the investigated 7 different chord types played with different chord voicings on various keyboard and guitar instruments.<sup>1</sup>

## 2. RELATED WORK

Early algorithms for acoustic modeling in ACR use template matching in chromagram representations, which encode the local saliency of different pitch classes in audio signals [1, 2]. Here, musical knowledge about the interval structures in different chord types is used to design chord templates for template matching algorithms. We refer the reader to [3] for a systematic overview over traditional techniques for feature extraction and pattern matching in ACR systems and the importance of pre-processing and post-processing steps.

In contrast, fully data-driven approaches based on deep neural network architectures have been lately shown to outperform hand-crafted feature representations. For instance, Convolutional Neural Networks (CNN) [4], Recurrent Neural Networks (RNN) [5, 6], and Feed-Forward Neural Networks (DNN) [7] are used as the acoustic modeling part. Most CNN-based approaches follow the VGG-style architecture [8] with a sequence of 2D convolutional layers and

Copyright: © 2019 Christon-Ragavan Nadar et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

<sup>1</sup>The dataset can be accessed at [https://www.idmt.fraunhofer.de/en/business\\_units/m2d/research.html](https://www.idmt.fraunhofer.de/en/business_units/m2d/research.html).

max pooling layers for a gradual down-sampling in the time-frequency space. Common time-frequency representations such as Short-time Fourier Transform (STFT) [9], Constant-Q transform (CQT) [4] or its multi-channel extension Harmonic CQT [10] are used as two-dimensional input to the CNN models.

As we focus on the acoustic modeling in ACR algorithms, we only briefly review temporal modeling techniques here. The first approaches for temporal modeling in ACR systems have used techniques from automatic speech recognition such as Hidden Markov models (HMMs) [11, 12]. Recently, Korzeniowski & Widmer use RNN-based chord language and duration models as post-processing after a CNN-based acoustic model [13]. Wu & Li combine a bi-directional Long Short-Term Memory (LSTM) network for sequence modeling and Conditional Random Field (CRF) to infer the final chord label sequence [10].

In real-life music recordings, the occurrence of different chord types is heavily imbalanced. While major and minor chords make up the bulk of annotated chords in available chord recognition datasets, other chord types such as seventh chords are heavily underrepresented. Hence, it becomes hard to train ACR systems to detect such chord types. If ACR algorithms should for instance be used to analyze jazz-related music styles, it becomes mandatory to extend the chord vocabulary by seventh chords. Only a few publications such as [10, 14–16] focus on extended-vocabulary chord recognition and go beyond the common 24 class major/minor chord vocabulary. In order to facilitate training models for the extended-vocabulary ACR, we created and published a novel dataset for large-scale chord recognition which will be detailed in Section 4.2.

### 3. SYSTEM OVERVIEW

#### 3.1 Input Features

Audio signals with a sample rate of 44.1 kHz are converted into Short-time Fourier Transform (STFT) magnitude spectrograms using a blocksize of 8192 (186 ms), a hopsize of 4410 (100 ms), and a Hann window. The phase is discarded. Using a triangular filterbank, the spectrogram is mapped to a logarithmically-spaced frequency axis with 133 frequency bins and a resolution of 24 bins per octave as in [9]. Logarithmic magnitude compression is used to increase the invariance to dynamic fluctuations in the music signal. Spectral patches are extracted with a blocksize of 15 (1500 ms) and a hopsize of 4 (400 ms) and fed as two-dimensional input to the CNN model.

#### 3.2 Modeling Strategies & Network Architecture

Figure 1 shows the CNN model architecture, which we adopted from [9]. As shown in Figure 2, we compare two modeling strategies for ACR: In the first strategy (**S1**), we aim to directly classify the chord label and use a single-output model. Depending on the chord vocabulary size, the final dense layer has either 24 units for classifying major & minor chords or 84 units for classifying all 7 chord types listed in Table 1 given all possible 12 chord root pitch classes. In the second strategy (**S2**), we predict the chord

Abbreviation	Chord Type (# Chord Tones)
5	“Power-chord” (2)
maj	Major chord (3)
min	Minor chord (3)
maj7	Major-seventh chord (4)
min7	Minor-seventh chord (4)
dom7	Dominant-seventh chord (4)
m7b5	Half-diminished seventh chord (4)

Table 1. Investigated chord types with the corresponding number of chord tones.

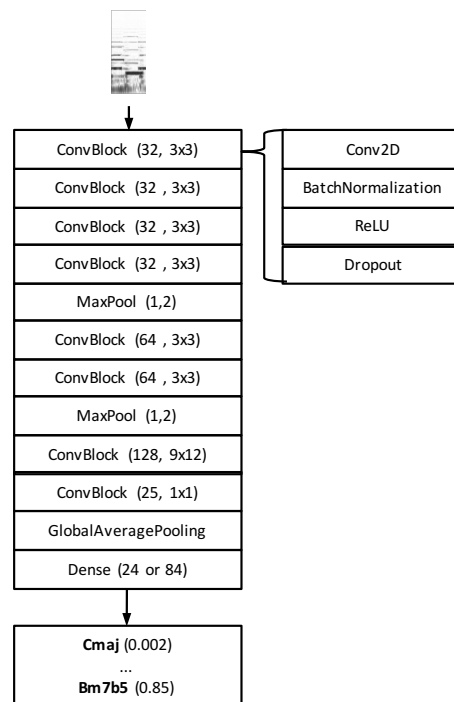


Figure 1. Architecture of the applied CNN. Number of filters and the kernel size are given in brackets for each ConvBlock. The softmax activation function is used in the final dense layer.

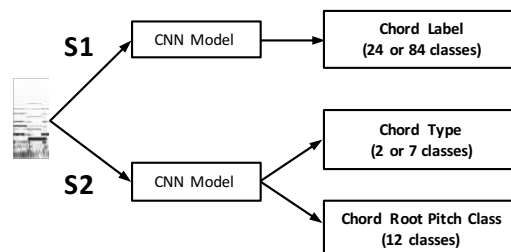


Figure 2. Illustration of two modeling strategies **S1** and **S2** for joint and separate chord type & root pitch class estimation.

root pitch class (12 classes) and chord type (2 or 7 classes) using two separate models. In both scenarios, the final dense layers have a softmax activation function and after

Dataset		# Files	Duration (h)	# Chord Segments
Bs	Beatles	1152	53.1	86868
Qn	Queen	180	11.2	20610
RW	Robbie Williams	234	19.1	25569
RWC	RWC	900	61.0	110331
Os	Osmalsky	7200	3.8	7200
Combi7	Combined Dataset	1863	112.2	193194
ISGuitar	IDMT_SMT_GUITAR	48	32.1	684
ISChords	IDMT_SMT_CHORDS	16	4.1	7398
ISInhouse	IDMT Inhouse Dataset	111	4.9	9159

Table 2. Overview of all chord recognition datasets with the respective number of audio files, the total duration in hours, as well as the number of chord segments.

each convolutional layer batch normalization [17] and a rectified linear unit (ReLU) activation function is applied. During training, we use the categorical cross-entropy loss, 500 training epochs with early stopping, the Adam optimizer [18] with a learning rate of 0.003, and a batch size of 256. The input features were normalized to zero-mean and unit-variance for the whole training set. The normalization values were later applied to the test data. All experiments were conducted using the Keras framework with Tensorflow as backend.<sup>2</sup>

## 4. DATASETS

### 4.1 Existing Datasets

The datasets used in this paper are summarized in Table 2. In addition to the total number of files, Table 2 provides the total dataset duration and total number of chord segments. In order to enlarge the dataset, we use pitch-shifting with total shifts of up to 4 semitones upwards and downwards as data augmentation technique. Hence, each original file results in 9 augmented files including the original recording. The datasets Beatles (Bs) [19], Queen (Qn) [19], Robbie Williams (RW) [20], RWC (100 songs from the RWC Popular Music Database [21]), and Osmalsky (Os) [22] have been used in the chord recognition literature previously. While the first four datasets include mixed music recordings with multiple instruments, the Os as well as the ISGuitar dataset (excerpts from the IDMT\_SMT\_GUITAR database published in [23]) consist of isolated recordings of different instruments playing chords. We created and published a novel dataset for chord recognition research (IDMT\_SMT\_CHORDS, abbreviated as ISChords in this paper), which will be detailed in the following section 4.2. The ISInhouse dataset is an in-house dataset covering various pop and rock music recordings, which cannot be published due to copyright constraints. In order to evaluate our model on music mixtures for the task of extended-vocabulary ACR, we aggregated an additional dataset (Combi7) using files which include seventh chord annotations from the datasets Bs, Qn, RWC, RW, and Os.

<sup>2</sup> Keras: [keras.io](https://keras.io), Tensorflow: [www.tensorflow.org](https://www.tensorflow.org)

Dataset	maj	min	maj7	min7	5	dom7	m7b5
Bs	67.95	20.49	2.17	3.00	0.04	6.12	0.22
Qn	63.81	22.52	1.28	4.47	1.28	6.56	0.09
RW	69.64	28.30	0.36	0.50	0.89	0.32	-
RWC	48.09	26.57	5.94	13.25	-	5.87	0.28
Os	60.00	40.00	-	-	-	-	-
Combi7	53.24	25.20	4.65	9.75	0.19	6.71	0.27
ISGuitar	67.89	16.51	4.59	3.67	-	5.50	1.83
ISChords	17.11	17.11	14.31	14.31	8.55	14.31	14.31
ISInhouse	62.85	37.15	-	-	-	-	-

Table 3. Chord type distribution per dataset in percent (%).

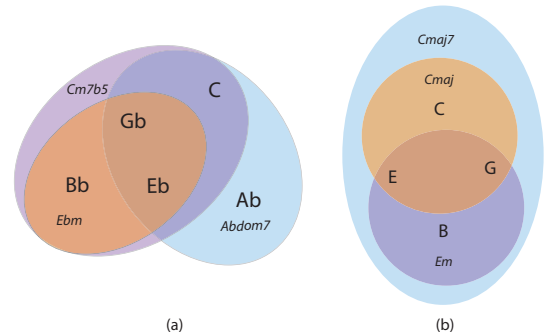


Figure 3. Illustration of ambiguities between chords due to shared chord tones between the chord types m7b5, min, and dom7 (a), and maj7, maj, and min (b). Figure inspired by [1].

### 4.2 Synthetic Dataset for Extended-Vocabulary Chord Recognition

Currently used chord recognition datasets are only partially suitable for training and evaluation on seventh chord types. Therefore, we created and published the novel IDMT\_SMT\_CHORDS dataset<sup>3</sup>. We initially created two MIDI files which cover all seven chord types listed in Table 1. Here we focused on chord voicings, which are commonly used on keyboard instruments and guitars. The piano MIDI file includes all chord types in all possible root note positions and inversions. The guitar MIDI file is based on barré chord voicings with the root note located on the low E, A, and D strings. We used several software instruments from Ableton Live<sup>4</sup> and Garage Band<sup>5</sup> to synthesize these MIDI files with various instruments such as piano, synthesizer pad, as well as acoustic and electric guitar.

## 5. EVALUATION

In the experiment described in this section, we focus on two types of ACR challenges: First, as discussed in [1], the assignment of a chord label is often ambiguous as different chord types partly share chord tones. Figure 3 illustrates these ambiguities for two chord types which share multiple chord tones. For instance, as shown on the left side, a half-diminished seventh chord (e.g., Cm7b5) can potentially be confused with different pitch classes like the

<sup>3</sup> The download link for the audio and MIDI files will be published in the camera-ready version of the paper.

<sup>4</sup> <https://www.ableton.com/en/live/>

<sup>5</sup> <https://www.apple.com/mac/garageband/>



C	84%	0%	2%	1%	1%	4%	0%	2%	2%	1%	1%	0%
C#	1%	84%	0%	2%	1%	2%	4%	0%	3%	2%	1%	1%
D	1%	1%	86%	0%	1%	1%	2%	3%	0%	2%	2%	1%
D#	2%	1%	1%	86%	0%	1%	1%	1%	3%	0%	2%	1%
E	2%	3%	2%	1%	82%	0%	2%	1%	2%	3%	0%	3%
F	2%	2%	3%	1%	0%	85%	0%	1%	2%	1%	3%	0%
F#	0%	3%	2%	3%	1%	1%	84%	0%	2%	1%	1%	3%
G	4%	0%	4%	2%	2%	2%	1%	81%	0%	1%	2%	1%
G#	1%	3%	0%	3%	1%	2%	1%	0%	85%	0%	1%	1%
A	2%	2%	4%	0%	2%	1%	3%	1%	1%	82%	0%	2%
A#	2%	1%	2%	3%	0%	4%	1%	2%	2%	0%	83%	0%
B	0%	2%	1%	2%	2%	0%	3%	1%	3%	1%	1%	83%
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B

Figure 4. Confusion matrix for chord root pitch class classification on isolated chord recordings (ISChords, experiment E3, strategy S2).

minor chord built upon its minor third ( $Ebm$ ) or with the dominant seventh chord built by introducing ( $Ab$ ) as a root note ( $Abdom7$ ).

Secondly, the datasets introduced in Section 4 have different acoustic characteristics. While some of the songs in the Bs and Qn datasets were recorded in the 1970s, other datasets such as RW and ISInhouse contain contemporary popular music recordings with a modern sound. Also, the datasets are of different timbre complexity ranging from simple isolated chords to complex audio mixtures. It was observed in related MIR tasks such as music transcription [24] that data-driven models trained for transcribing isolated notes do not generalize well to more complex acoustic mixtures. Here, we aim to investigate whether such findings can be replicated for ACR.

Table 4 summarizes 11 experiments, which are designed to analyze the chord type ambiguity on isolated chord recordings (E1 - E3, see Section 5.1), the generalization of ACR models to mixture recordings (E4 - E6, see Section 5.2), as well as two real-life ACR application scenarios (E7 - E11, see Section 5.3). In addition, we tested the state-of-the-art ACR algorithm proposed in [9] as our reference system (REF) for the major/minor chord vocabulary (24 classes). The implementation from the madmom [25] python library was used and its performance is documented in the last column of Table 4.

In all experiments, audio recordings are split into training and test set on a dataset-level or on a file-level. When a dataset is used for training and test we perform a two-fold random cross-validation. We use the weighted average class f-score throughout this paper as evaluation measure. The f-scores  $F_{24}$  and  $F_{84}$  are used to indicate if the evaluation was performed on 24 chord classes (major/minor vocabulary) or 84 classes (extended-vocabulary ACR). The “no chord” class is neglected in all experiments. In the following subsections, three groups of experiments will be detailed whose results are summarized in Table 4.

maj	90%	0%	6%	0%	4%	0%	0%
min	0%	81%	8%	7%	3%	0%	0%
maj7	3%	4%	89%	0%	3%	0%	0%
min7	0%	1%	0%	98%	1%	0%	0%
5	15%	0%	0%	0%	85%	0%	0%
dom7	2%	1%	4%	2%	0%	87%	5%
m7b5	0%	0%	1%	3%	1%	0%	95%
	maj	min	maj7	min7	5	dom7	m7b5

Figure 5. Confusion matrix for 7 chord types in extended-vocabulary ACR on isolated chord recordings (ISChords, experiment E3, strategy S2).

maj	86%	4%	2%	2%	0%	6%	0%
min	10%	75%	2%	10%	0%	3%	0%
maj7	48%	6%	31%	10%	0%	3%	1%
min7	14%	15%	1%	64%	0%	6%	0%
5	13%	3%	0%	1%	79%	4%	0%
dom7	49%	5%	1%	3%	0%	42%	0%
m7b5	13%	37%	0%	5%	0%	41%	3%
	maj	min	maj7	min7	5	dom7	m7b5

Figure 6. Confusion matrix for 7 chord types in extended-vocabulary ACR on mixed chord recordings (Combi7 + ISChords, experiment E6, strategy S2).

## 5.1 Chord Type Ambiguity on Isolated Chord Recordings

In experiments E1, E2, and E3 (first section of Table 4), we train and evaluate ACR models on isolated chord recordings (ISChords) to study the effect of chord tone ambiguity in extended-vocabulary ACR. As explained in Section 4.2, the contained chords are based on two systematically generated MIDI files with chord voicings from keyboard and non-keyboard instruments. In our experiments, we evaluate the influence of the chord voicing types as well as of the modeling approach (compare Section 3.2).

For the major/minor chord vocabulary (24 classes), we obtain high f-scores  $F_{24}$  between 0.81 and 0.99 using the strategy S1. In the two experiments E1 & E2, we perform a chord voicing “cross-test” by exclusively assigning piano chord voicings to the training set and test on non-piano chord voicings and vice versa. Intuitively, we observe lower f-scores (compared to E3) since the models are confronted with a different timbre (instrument) and previously unseen chord voicings at test time. Contrary to the 24 classes major/minor scenario, we observe that for the 84 classes scenario (extended-vocabulary ACR),

#	Training Set	Test Set	Strategy S1		Strategy S2		Reference System (REF)
			$F_{24}$	$F_{84}$	$F_{24}$	$F_{84}$	$F_{24}$
Chord Type Ambiguity on Isolated Chord Recordings (Section 5.1)							
E1	ISChords (non-guitar)	ISChords (guitar)	0.92	0.58	0.90	0.76	0.74
E2	ISChords (guitar)	ISChords (non-guitar)	0.81	0.49	0.54	0.56	0.71
E3	ISChords	ISChords	0.99	0.97	0.90	0.82	0.75
Generalization of ACR Models towards Complex Recordings (Section 5.2)							
E4	ISChords	Combi7	0.40	0.36	0.18	0.28	0.83
E5	Combi7	Combi7	0.83	0.63	0.84	0.64	0.83
E6	ISChords + Combi7	ISChords + Combi7	0.84	0.65	0.84	0.66	0.81
Real-Life ACR Application Scenarios (Section 5.3)							
E7	ISChords	ISInhouse	0.56	-	0.27	-	0.76
E8	ISChords	ISGuitar	0.90	-	0.70	-	0.91
E9	Bs + Qn + RW + RWC + Os + ISChords	ISInhouse	0.71	-	0.74	-	0.76
E10	Bs + Qn + RW + RWC + Os + ISChords	ISGuitar	0.90	-	0.91	-	0.91
E11	Bs + Qn + RW + RWC + Os + ISChords	Bs + Qn + RW + RWC + Os + ISChords	0.81	-	0.84	-	0.78

Table 4. This table lists all ACR experiments grouped into three sections described in Section 5.1, Section 5.2, and Section 5.3. For each experiment, the second and third column introduce the applied training set and test set. For both modeling strategies **S1** and **S2** introduced in Section 3.2, f-scores  $F_{24}$  and  $F_{84}$  are provided for the 24 classes major/minor chord vocabulary and the 84 classes extended-vocabulary with the 7 chord types as listed in Table 1. For each experiment, the best scores for each of the vocabulary are highlighted using bold font. The last column shows the f-score using the reference system (**REF**) on the test set.

strategy **S2** clearly outperforms **S1**. We assume that the network capacity is large enough to learn distinct spectral patterns for classifying among 24 chord labels. For the extended-vocabulary scenario however, the amount of 84 classes is presumably too high to be learnt by one model using strategy **S1**. Instead, splitting the classification task into two easier sub-tasks (with not more than 12 classes each) using strategy **S2** seems slightly beneficial here. Interestingly, in experiment **E3**, where all chord voicings are mixed, strategy **S1** outperforms strategy **S2** in both the 24 and 84 classes scenarios. When testing with state of the art model (**REF**) in **E3** we see that **REF** does not perform as well since it is likely trained on complex audio mixtures.

Figure 4 shows the confusion matrix for the classification of the chord root pitch class for the 84 class scenario for experiment **E3**. It can be observed that the model shows a good performance for all classes between 82 % and 86 %. Similarly, as can be seen in Figure 5, the model easily learns to distinguish between different chord shapes for isolated chord recordings (ISChords dataset). However, Figure 6 shows the more complicated test case of mixed audio recordings (Combi7 + ISChords datasets). The most prominent misclassifications between the maj7 towards the maj, the dom7 towards the maj, as well as the m7b5 towards the min and the dom7 all confirm the chord tone ambiguities discussed in Section 5.

## 5.2 Generalization of ACR Models towards Complex Recordings

In experiments **E4** to **E6** (second section of Table 4), we investigate (similar to [24]) whether and to what extent ACR models trained on isolated instrument recordings generalize towards complex music recordings in the Bs, Qn, and RWC datasets. Also, we test whether adding the proposed ISChords dataset can help to improve the performance on extended-vocabulary ACR. As expected, a poor f-score of  $F_{24} = 0.4$  in **E4** shows that the investigated CNN-based ACR model does not generalize well from a simple training scenarios (ISChords) towards a complex test scenario (Combi7). The clearly higher f-scores of  $F_{24} = 0.84$  and  $F_{84} = 0.66$  show that this kind of data-driven classification models need to be trained on data of similar timbre complexity as in the test scenario. We only observe a small improvement of 0.02 (from **E5** to **E6**) for the 84 classes scenario in f-score when training with both datasets (**E6**). The reference algorithm **REF** performs similar to **S2** in **E5** and slightly worse than **S1** and **S2** with a difference of 0.03 in **E6**.

## 5.3 Real-Life ACR Application Scenarios

In the experiments **E7** to **E11** (third section of Table 4), we address realistic requirements for ACR systems to be deployed in real-life applications. In a music education scenario, musical instruments usually can be directly recorded and analyzed without background sounds. Therefore, we test the chord recognition performance on isolated poly-

phonic electric guitar recordings (ISGuitar), which include both chords and arpeggios. In a music annotation scenario, we evaluate ACR models on a set of 111 contemporary pop and rock music recordings of various instrumentations (ISInhouse). Similarly to **E4**, we can observe in experiment **E7** that ACR models trained only on isolated chord recordings do not perform well on complex mixtures (ISInhouse). However, such models show a good performance (**S1**,  $F_{24} = 0.9$ , **S2**,  $F_{24} = 0.7$ ) when being applied to isolated guitar recordings (**E8**). In both test cases, the performance can be clearly improved by adding more datasets to the training set, which reflect a larger variety of music recordings (compare experiments **E9** and **E10**). In both experiments **E9** and **E10**, the reference algorithm **REF** performs almost similar except for **E11** where **S2** achieves a slightly better f-score.

## 6. CONCLUSIONS

In this paper, we used a state-of-the-art Deep Convolutional Neural Network for ACR. In addition to publishing a novel dataset of isolated chord recordings, we propose an alternative modeling strategy using two models for the separate classification of the chord type and the chord root pitch class. In our experiments, we first evaluate this strategy for the controlled test case of isolated instrument recordings. Most of the chord type misclassifications are due to shared chord tones. The results indicate that ACR even with extended-vocabulary is feasible (f-scores above 0.9), but the performance depends on whether the chord voicings and instrument timbre used in the test set have been learnt by the model before.

In a second set of experiments, we were able to replicate the finding from automatic music transcription that data-driven ACR models need to be trained on data of the same complexity as the expected test data. Models trained on isolated instrument recordings performed poorly on mixed audio data. Finally, we evaluated the CNN model on two separate datasets, which acted as a proxy for deploying an ACR model into real-life production systems for the two use cases music education and music annotation. Here, we achieved high f-scores of 0.91 for isolated guitar recordings and 0.74 for mixed contemporary popular music recordings showing the usefulness for real-life MIR applications.

## Acknowledgments

This work has been supported by the German Research Foundation (AB 675/2-1, BR 1333/20-1).

## 7. REFERENCES

- [1] M. Müller, *Fundamentals of Music Processing*. Springer Verlag, 2015.
- [2] T. Fujishima, “Real-time chord recognition of musical sound: A system using common lisp music,” in *Proceedings of the 1999 International Computer Music Conference (ICMC)*, Beijing, China, 1999, pp. 464–467.
- [3] T. Cho and J. P. Bello, “On the relative importance of individual components of chord recognition systems,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 477–492, 2014.
- [4] E. J. Humphrey and J. P. Bello, “Rethinking automatic chord recognition with convolutional neural networks,” in *Proceedings of the 11th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando, FL, USA, 2012, pp. 357–362.
- [5] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, “Audio chord recognition with recurrent neural networks,” in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, 2013, pp. 335–340.
- [6] S. Sigtia, N. Boulanger-Lewandowski, and S. Dixon, “Audio chord recognition with a hybrid recurrent neural network,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, 2015, pp. 127–133.
- [7] X. Zhou and A. Lerch, “Chord detection using deep learning,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, 2015, pp. 52–58.
- [8] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.
- [9] F. Korzeniowski and G. Widmer, “A fully convolutional deep auditory model for musical chord recognition,” in *Proceedings of the 26th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Salerno, Italy, 2016, pp. 1–6.
- [10] Y. Wu and W. Li, “Automatic audio chord recognition with midi-trained deep feature and blstm-crf sequence decoding model,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 355–366, 2019.
- [11] A. Sheh and D. P. Ellis, “Chord segmentation and recognition using em-trained hidden markov models,” in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, Baltimore, Maryland, USA, 2003.
- [12] J. Deng and Y.-K. Kwok, “Large vocabulary automatic chord estimation using deep neural nets: Design framework, system variations and limitations,” *arXiv preprint arXiv:1709.07153*, 2017.
- [13] F. Korzeniowski and G. Widmer, “Improved chord recognition by combining duration and harmonic language models,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, 2018, pp. 10–17.

- [14] M. Mauch and S. Dixon, "Approximate note transcription for the improved identification of difficult chords," in *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, Utrecht, Netherlands, 2010, pp. 135–140.
- [15] J. Deng and Y. Kwok, "Automatic chord estimation on seventhsbass chord vocabulary using deep neural network," in *Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, China, 2016, pp. 261–265.
- [16] S. Gasser and F. Strasser, "Mirex 2018: Multi objective chord estimation," 2018, (last accessed 28.03.2019). [Online]. Available: <https://www.music-ir.org/mirex/abstracts/2018/SG1.pdf>
- [17] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.
- [18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.
- [19] "Isophonics dataset reference annotations," (last accessed 24.01.2019). [Online]. Available: <http://isophonics.net/datasets>
- [20] B. Di Giorgi, M. Zanoni, A. Sarti, and S. Tubaro, "Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony," in *Proceedings of the 8th International Workshop on Multidimensional Systems (nDS)*, Erlangen, Germany, 2013, pp. 1–6.
- [21] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical and jazz music databases," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002, pp. 287–288.
- [22] J. Osmalsky, V. D. M. Embrechts, Jean-Jacques, and S. Pierard, "Neural networks for musical chords recognition," *Journées d'informatique musicale*, pp. 39–42, 2012.
- [23] "IDMT SMT GUITAR dataset," (last accessed 24.01.2019). [Online]. Available: [https://www.idmt.fraunhofer.de/en/business\\_units/m2d/smt/guitar.html](https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/guitar.html)
- [24] J. Abeßer, S. Balke, and M. Müller, "Improving bass saliency estimation using label propagation and transfer learning," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, 2018, pp. 306–312.
- [25] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "madmom: a new Python Audio and Music Signal Processing Library," in *Proceedings of the 24th ACM International Conference on Multimedia*, Amsterdam, The Netherlands, 2016, pp. 1174–1178.



# From Jigs and Reels to Schottisar och Polskor: Generating Scandinavian-like Folk Music with Deep Recurrent Networks

Eric Hallström   Simon Mossmyr   Bob L. Sturm   Victor Hansjons Vegeborn   Jonas Wedin

KTH Royal Institute of Technology - Speech, music and hearing division

{erichal, mossmyr, bobs, victorhv, jonwed}@kth.se

## ABSTRACT

The use of recurrent neural networks for modeling and generating music has been shown to be quite effective for compact, textual transcriptions of traditional music from Ireland and the UK. We explore how well these models perform for textual transcriptions of traditional music from Scandinavia. This type of music has characteristics that are similar to and different from that of Irish music, e.g., mode, rhythm, and structure. We investigate the effects of different architectures and training regimens, and evaluate the resulting models using three methods: a comparison of statistics between real and generated transcriptions, an appraisal of generated transcriptions via a semi-structured interview with an expert in Swedish folk music, and an exercise conducted with students of Scandinavian folk music. We find that some of our models can generate new transcriptions sharing characteristics with Scandinavian folk music, but which often lack the simplicity of real transcriptions. One of our models has been implemented online at <http://www.folkrnn.org> for anyone to try.

## 1. INTRODUCTION

Recent work [1] applies long short-term memory (LSTM) neural networks [2] to model and generate textual transcriptions of traditional music from Ireland and the UK. The data used in that work consists of over 23,000 tune transcriptions crowd-sourced online.<sup>1</sup> Each transcription is expressed using a compact textual notation called ABC.<sup>2</sup> The resulting transcription models have been used and evaluated in a variety of ways, from creating material for public concerts [3] and a professionally produced album [4], to numerical analyses of the millions of parameters in the network [5, 6], to an accessible online implementation.<sup>3</sup> The success of machine learning in reproducing idiosyncrasies of Irish traditional music transcriptions comes in large part from the expressive capacity of the LSTM network, the

compact data representation designed around ABC notation, and a large amount of training data. Will such a model also perform well given another kind of traditional music expressed in a similarly compact way? What happens when the amount of training data is an order of magnitude less than for the Irish transcription models?

In this paper, we present our work applying deep recurrent modeling methods to Scandinavian folk music. We explore both LSTM and Gated Recurrent Unit (GRU) networks [7], trained with and without dropout [8]. We acquire our data from a crowd-sourced repository of Scandinavian folk music, which gives 4,083 transcriptions expressed as ABC notation. Though this data is expressed the same way as the Irish transcriptions used in [1], there are subtle differences between the styles that require a different approach, e.g., key changes in tunes. This results in a larger vocabulary for the Scandinavian transcription models, compared with the Irish ones (224 vs. 137 tokens) [1]. We also explore using pretraining with the Irish transcription dataset, with further training using only Scandinavian transcriptions. To evaluate the resulting models, we compare low-level statistics of the generated transcriptions with the training data, conduct a semi-structured interview with an expert on Swedish folk music, and perform an exercise with students of Scandinavian folk music.

We begin by briefly reviewing recurrent neural networks, including LSTM and GRU networks. We then describe the data we use, how we have process it to create training data, and how we train our models. We then present our evaluation of the models, and discuss the results and our future work.

## 2. RECURRENT NEURAL NETWORKS

A Recurrent Neural Network (RNN) [9] is a type of artificial neural network that uses directed cycles in its computations, inspired by the cyclical connections between neurons in the brain [10]. These recurrent connections allow the RNN to use its output in a sequence, while the internal states of the network act as memory. We test two different flavors of RNN: Long Short-Term Memory Networks (LSTM), and Gated Recurrent Units (GRU). The final layer of these networks is a softmax layer, which produces a conditional probability distribution over a vocabulary given the previous observations. It is from this distribution one samples to generate a sequence.

<sup>1</sup> <http://thesession.org>

<sup>2</sup> <http://abcnotation.com/wiki/abc:standard:v2.1>

<sup>3</sup> <http://www.folkrnn.org>

## 2.1 Long Short-Term Memory (LSTM)

The LSTM is an RNN architecture designed to overcome problems in training conventional RNNs [2]. Each LSTM layer is defined by four “gates” transforming an input  $x_t$  at time step  $t$  and a previous state  $h_{t-1}$  as follows [11]:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (2)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$

$$c_t = i_t \odot \tanh(W_u x_t + U_u h_{t-1} + b_u) + f_t \odot c_{t-1} \quad (4)$$

where  $\sigma()$  denotes the element-wise logistic sigmoid function, and  $\odot$  denotes the element-wise multiplication operator. The LSTM layer updates its hidden state by

$$h_t = o_t \odot \tanh(c_t). \quad (5)$$

The hidden state of an LSTM layer is the input to the next deeper layer.

## 2.2 Gated Recurrent Unit (GRU)

A GRU layer is similar to that of the LSTM, but each layer uses only two gates and so is much simpler to compute [7]. Each GRU layer transforms an input  $x_t$  and a previous state  $h_{t-1}$  as follows:

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad (6)$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z). \quad (7)$$

The GRU layer updates its state by

$$h_t = (1 - z_t) \odot \tanh(W_h x_t + U_h (r_t \odot h_{t-1})) + z_t \odot h_{t-1}. \quad (8)$$

Compared with the LSTM, each GRU layer has fewer parameters.

## 3. MODELING SCANDINAVIAN FOLK MUSIC

### 3.1 Data

FOLKWiki<sup>4</sup> is a wiki-style site dedicated to Scandinavian folk music that allows users to submit tune transcriptions to a growing database, each expressed using ABC notation. We collect transcriptions from FOLKWiki by using a web scraper,<sup>5</sup> recursively gathering them using the “key” category.<sup>6</sup> This produces 4083 unique transcriptions. An example transcription is shown in the following:

```
%abc-charset utf-8
```

```
X:1
T:Visa
T:ur Svenska Folkmelodier
  utgivna av C.E. Södling
B:http://www.smus.se/... (Edited by authors)
O:Småland
```

<sup>4</sup> <http://www.folkwiki.se>

<sup>5</sup> <http://www.scrapy.org>

<sup>6</sup> <http://www.folkwiki.se/Tonarter/Tonarter>

```
N:Se även +
M:3/4
L:1/8
R:Visa
Z:Nils L
K:Am
EE A2 cc | ee B2 d2 | cB (Ac) BA | ^G2 E4 ::
w:ung-er-sven med ett hur-tigt mod han
  sving-ar sig * u-ti la-get
EE A2 B2 | cd e2 d2 | cB Ac B^G | A2 A4 :|
w:fem-ton al-nar grö-na band det bär han
  u-ti sin skjort-kra-ge
```

We process these transcriptions in the following way:

1. Remove all comments and non-musical data
2. If the tune has multiple voices, separate them as if they are individual tunes
3. Parse the head of the tune and keep the length (L:), meter (M:), and key fields (K:)
4. Parse the body of the tune
5. Clean up and substitute a few resulting tokens to keep similarity over the data set (i.e. “K:DMajor” is substituted by “K:DMaj” etc.)

We keep all the following tokens in the tunes body:

- Changes in key (K:), meter (M:) or note length (L:)
- Any note as described in the ABC-Standard (e.g., e, =a or any valid note)
- Duplets (2, triplets (3, quadruplets (4, etc.
- Note length (Any integer after a note =a 4)
- Rest sign (z)
- Bars and repeat bars (: | | :)
- Grouping of simultaneous notes ([ and ])

After processing, the transcription above appears as:

```
[L:1/8]
[M:3/4]
[K:AMin]
EEA2cc|eeB2d2|cBAcBA|
^G2E4::|:EEA2B2|cde2d2|
cBAcB^G|A2A4::|
```

Each symbol separated by a space corresponds to one token in the model vocabulary. Notice that almost all meta-data fields are removed, as well as lyrics. Reprise bars such as :: or :|: have been substituted by :| |: to minimize the vocabulary size so the models become less complex. The output produced by our text processing is a file with all transcriptions separated by a newline. We do not keep any transcriptions with fewer than 50 tokens or more than 1000 tokens. We also do not attempt to correct human errors in transcription (e.g., miscounted bars). The resulting dataset is available in a repository.<sup>7</sup> The parser we created to do the above is available at the project repository.<sup>8</sup> The total number of unique tokens in the Folkwiki dataset is 155.

<sup>7</sup> [https://github.com/victorwegeborn/folk-rnn/tree/master/data/9\\_nov](https://github.com/victorwegeborn/folk-rnn/tree/master/data/9_nov)

<sup>8</sup> <http://www.github.com/ztime/polska>

### 3.2 Pretraining models

The training of deep models typically begins with a random initialization of its weights, but it can also begin with weights found from previous training. In the latter sense, one can think of it as making the network first aware of syntactical relationships in the domain in which it is working, and then tuning the network on a subset to specialize it. We experiment with training models first using the concatenation of the FolkWiki dataset with the Irish transcription dataset that we process in the same way,<sup>9</sup> and then tuning the model with just the FolkWiki dataset.

### 3.3 Dropout

A danger with machine learning in general is the tendency to overfit to training data. One method to prevent overfitting of a network is to use a mechanism called dropout [8]. Dropout works by masking the output of a layer in the network with a random distributed binary vector during training. The dropout probability  $p_i$  is the parameter of the model that decides what output of the layer is propagated. When we use dropout, we set  $p_i = 0.5$ .

### 3.4 Model architecture and training

We use two different neural networks based on the LSTM and GRU units, with three different variations:

- LSTM with 50% dropout trained on FolkWiki ( $L_{50}^F$ )
- GRU with 50% dropout trained on FolkWiki ( $G_{50}^F$ )
- LSTM with 50% dropout pretrained on FolkWiki and TheSession, then only FolkWiki ( $L_{50}^{S+F}$ )
- GRU with 50% dropout pretrained on FolkWiki and TheSession, then only FolkWiki ( $G_{50}^{S+F}$ )
- LSTM without dropout pretrained on FolkWiki and TheSession, then only FolkWiki ( $L^{S+F}$ )
- GRU without dropout pretrained on FolkWiki and TheSession, then only FolkWiki ( $G^{S+F}$ )

Because of the number of unique tokens for a model depends on its training data, we adjusted the number of hidden units in layers to be about 4 times the vocabulary size [6]. We trained two models on only FolkWiki using 600 hidden nodes in each layer, whilst the models trained on both session-data and FolkWiki used 800 hidden nodes (the vocabulary size of the concatenation of the two datasets is 224). During the pretraining phase we use a batch size of 64, and for the final training we use a batch size of 32. We use a learning rate  $\eta = 0.003$  with a decay of 0.97 after every 20 epochs (the same as used in [1]). All models have gradient clipping set to 5.

Figure 1 shows the mean transcription validation negative log-likelihood loss for each model. We train all models for 40 epochs on FolkWiki. The pretraining of the LSTM model on the FolkWiki and TheSession data is done for 50 epochs, but the pretraining of the GRU model on FolkWiki and TheSession data is done for 20 epochs.

<sup>9</sup> See footnote 7.

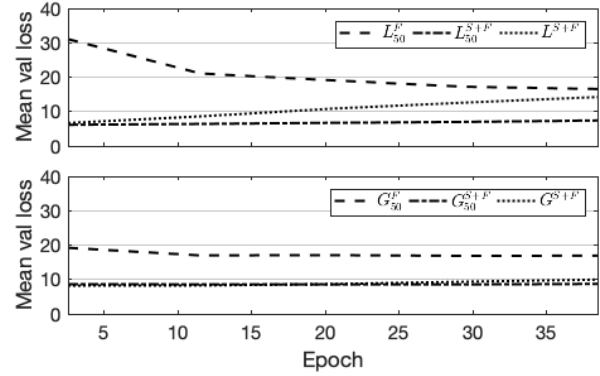


Figure 1. The mean transcription validation loss for the LSTM models (top) and the GRU models (bottom) when training on the FolkWiki dataset.

## 4. EVALUATION

We now evaluate the six different models we have trained. We use three different approaches to evaluation. First, we compare the descriptive statistics of the generated and real transcriptions. Second, we select output generated by the models for evaluation by an expert on Swedish traditional music. Finally, we perform an exercise with students of Scandinavian folk music.

### 4.1 Statistical analysis

We have each model generate 4000 transcriptions at random, and then look at how these compare with the 4083 transcriptions in the training dataset. Figure 2 compares FolkWiki with the transcriptions generated by  $L_{50}^F$  and  $G_{50}^F$  in terms of occurrences of keys, meter and number of tokens. We see a strong bias of  $G_{50}^F$  towards generating the D minor token, and away from the D major token, while  $L_{50}^F$  has a slight bias towards generating the tokens of D minor and G major, and away from D major. When it comes to the meter tokens,  $L_{50}^F$  appears to be in agreement with FolkWiki, while  $G_{50}^F$  is biased to most often produce 3/4. When it comes to the lengths of the transcriptions,  $L_{50}^F$  generates slightly shorter transcriptions than those in FolkWiki, while  $G_{50}^F$  generates transcriptions that are longer.

Figure 3 compares FolkWiki with the transcriptions generated by  $L_{50}^{S+F}$  and  $G_{50}^{S+F}$ . We see  $L_{50}^{S+F}$  is biased toward producing the D minor token and away from the D major and A minor tokens, while  $G_{50}^{S+F}$  too often generates the A major, D minor and G major tokens. As with the meters, these models either favor the 3/4 or the 4/4 tokens. In terms of number of tokens in the transcriptions,  $G_{50}^{S+F}$  generates longer ones than  $L_{50}^{S+F}$ , but both tend to produce longer transcriptions than in the FolkWiki dataset.

### 4.2 Semi-structured interview with Swedish expert

In order to learn about some defining characteristics of Swedish folk music, and to gauge the plausibility of material generated by our models, we interviewed Olof Mising, a lecturer in Music Theory and lecturer of folk violin

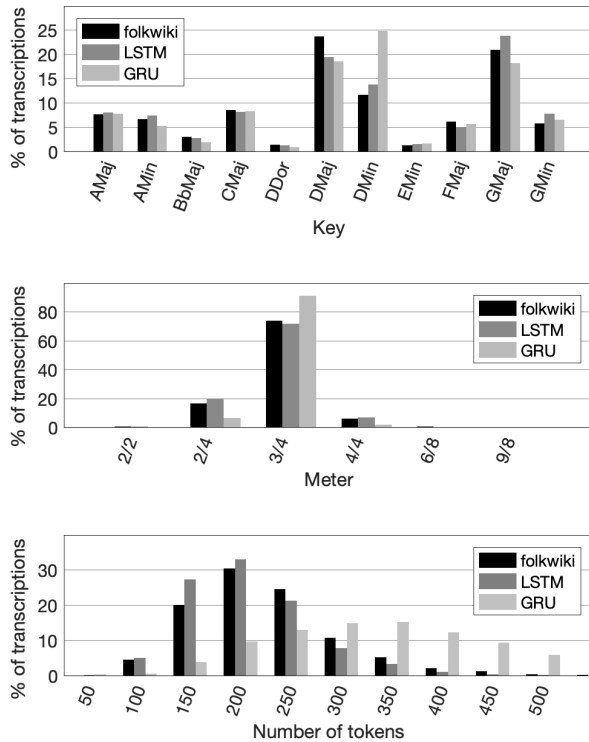


Figure 2. Comparison between FolkWiki and the  $L_{50}^F$  and  $G_{50}^F$  models. Percent of transcriptions in terms of keys (top), meters (middle), and number of tokens (bottom).

at the Royal College of Music (KMH) in Stockholm, Sweden. We told Misgeld about our research and why we were interested in interviewing him.

In preparation for the interview we created three different collections of transcriptions, each containing 500: 400 transcriptions generated by a model, and 100 real transcriptions from FolkWiki, randomly selected and ordered. The generated transcriptions of one collection come from  $L_{50}^{S+F}$  because its statistics most closely resemble the training data. In the second collection we chose to use transcriptions generated by  $L^{S+F}$  to see if not using dropout affects quality. Finally, for the third collection we chose transcriptions generated by  $G_{50}^F$  because its statistics look the most poor with respect to FolkWiki.

Misgeld first assessed collection  $L_{50}^{S+F}$ . We described the transcriptions “computer generated”, without mentioning that some of them were from FolkWiki. We asked him to freely browse through the collection and provide observations. After a few observations we asked him to find a transcription that is really good (in your opinion) and describe why, and to find a transcription that is really poor (in your opinion) and describe why. After more observations we told him that some of the transcriptions are from the training data (real tunes), and asked if he can locate them. We performed the same procedure with the other two collections. After this, we asked Misgeld to freely compare the three collections.

Misgeld identified distinct styles in the transcriptions generated by  $L_{50}^{F+S}$ . Some comments include “maybe Slång-

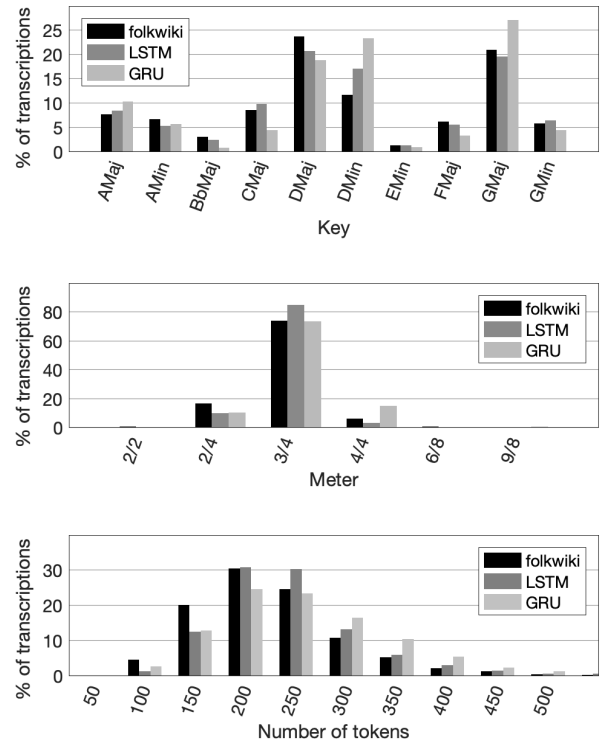


Figure 3. Comparison between FolkWiki and the  $L_{50}^{S+F}$  and  $G_{50}^{S+F}$  models. Percent of transcriptions in terms of keys (top), meters (middle), and number of tokens (bottom).

polska because of the 16th notes”, “like triplet Polska, very occupied with G”. When asked to find an example of a really good and a really bad transcription, both choices come from the generated material. Two of these are shown in Figs. 4 and 5.

For transcriptions generated by  $L^{F+S}$ , Misgeld comments that many generated tunes are “strange”, in one form or another, e.g., “strange jump”, “strange note”, “strange [in general]”, and “strange rhythm”. At the same time he felt transcriptions by this model were the most convincing. Our later analysis of the transcriptions generated by this model reveals it to be plagiarizing, which could explain Misgeld’s observation.

For the collection generated by  $G_{50}^F$ , Misgeld notes the transcriptions seem longer and more varied in structure. While he only gave specific comments on four transcriptions, two of which were generated, he notes that the generated transcriptions contain unusual chromaticism, no closure in rhythm, too many notes in a bar, and strange rhythmic patterns.

When asked for observations about all three models, and about the exercise, Misgeld said that “it’s interesting, ...it makes you curious how these models work.”, “The generated tunes appear to have too many ideas”, “no strong motif”, “not enough repetition and variation”, and “funny endings”. He later explained that traditional tunes often have simple and clear ideas, which assists the oral transmission of the tradition.





Figure 4. A tune generated by  $L_{50}^{S+F}$  for which the expert commented, “Seems real, natural ending, repeated with a 4+6 structure”.



Figure 5. A tune generated by  $L_{50}^{S+F}$  for which the expert commented, “Fragmented, unexpected. Not real.”

### 4.3 Exercise with folk music students

One of the authors (Sturm) was invited to give a workshop about AI and music to interested students of a folk music school in Bollnäs, Sweden. During one hour of the workshop, two groups of students were given different transcriptions and told to label each one as real or fake. Among the ten transcriptions given to each group, six were randomly generated by  $L_{50}^{S+F}$ , and four were randomly selected from the FolkWiki dataset. Points were awarded to each group based on the following: 2 points for each real transcription identified as real; 1 point for each fake transcription identified as fake; -1 point for each real transcription misidentified as fake; and -2 points for each fake transcription misidentified as real. The musicians were told they could also play the melodies as part of the evaluation.

One group decided to label everything as fake, and so received a total of 2 points. The other group was more deliberate, identified all real transcriptions as real, but misidentified three fake transcriptions as real, and so received a total of 5 points. Figure 6 shows the three transcriptions misidentified as real, which also illustrate some of the idiosyncrasies and weaknesses of the model. Transcription A has a pickup to the first bar which is not accounted for, but such a thing also occurs in the training data. That transcription A is so short added to the uncertainty of the students, as well as the lack of strong relationship between the two parts. Transcription B has a similar weak connection between the parts, but the first part is persuasive. The

students noticed the second part becomes somewhat stuck on E minor, but they felt it could be due to someone trying to be clever. The students felt transcription C could be fake, but also felt the relationship between its two parts was good.

Figure 7 shows the three fake transcriptions the students identified correctly. The students noted nothing was especially wrong in these transcriptions. They called transcription E ‘quirky’, but noted the ending of both parts does not make sense. They also noted that the second part of transcription F feels stuck.

## 5. DISCUSSION

By comparing some of the descriptive statistics of collections of transcriptions, real and generated, we can see that the models have learned some aspects of the transcriptions of Scandinavian music. We find that the LSTM models provide a better fit to the data than GRU models, with the latter creating on average longer transcriptions. Our semi-structured interview with an expert of Swedish folk music shows many more details about the success and failure of our models. The expert identified several characteristics of the generated transcriptions, e.g., that they seem to be unfocused, to have too many ideas, but that some can be quite plausible. Transcriptions produced by the LSTM model trained without dropout were the most convincing, but this is likely due to the fact that the model was plagiarizing large amounts of the training data. The expert also noted



Figure 6. Three transcriptions generated by  $L_{50}^{S+F}$  that students assessed as being real.

that the transcriptions generated by the GRU model were more modern and “adventurous” than the others, but not as plausible. Conducting an exercise with students of Scandinavian folk music provided other observations about the transcriptions, and how one may think about a melody being good or not. One observation was that some of the generated transcriptions do not end on the tonic. This could be due to the fact that in creating our dataset we separated multivoice transcriptions into multiple single-voice transcriptions. In such a case, the harmonizing voice becomes a melody which often ends on the third.

## 6. CONCLUSIONS

We have shown that deep recurrent networks can generate music transcriptions that share characteristics with those of Scandinavian folk music. Even though our dataset is about one-sixth the size of the dataset used to train previous models of Irish traditional music [1], we have shown that

the two datasets can be combined to pretrain a network, and then fine tune the network on the smaller Scandinavian music dataset to generate convincing transcriptions.

## Acknowledgments

**Google LLC** through the *Google Cloud Education Grant*, for providing us with credits to be used on their Google Cloud platform; **Olof Misgeld**, Lecturer in Music Theory with specialization in Swedish folk music, and Director of Studies, at the Institute for Folk Music at the Royal College of Music (KMH), Stockholm, Sweden; **Bollnäs Folkhögskola** and **Esbjörn Wettermark**.

## Author contribution

This work started from a data science project course at the Royal Institute of Technology in Stockholm, Sweden (Bob Sturm supervising). Apart from the joint effort in writing this paper the fine grained details of contributions from the



Figure 7. Three tunes generated by  $L_{50}^{S+F}$  that students assessed as being not real.

authors are the following; Eric Hallström managed the dependencies for Folk RNN, Simon Mossmyr modified the existing code base to use GRU layers instead of LSTM layers, Victor Hansjons Vegeborn assisted with the parser code and created code used for analysis and Jonas Wedin created the parser and scraper code.

## 7. REFERENCES

- [1] B. L. Sturm, J. F. Santos, O. Ben-Tal, and I. Korshunova, “Music transcription modelling and composition using deep learning,” in *Proc. Conf. Computer Simulation of Musical Creativity*, Huddersfield, UK, 2016.

- [2] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [3] B. L. Sturm, O. Ben-Tal, U. Monaghan, N. Collins, D. Herremans, E. Chew, G. Hadjeres, E. Deruty, and F. Pachet, “Machine learning research that matters for music creation: A case study,” *J. New Music Research*, vol. 48, no. 1, pp. 36–55, 2018.
- [4] B. L. Sturm and O. Ben-Tal, “Let’s have another Gan Airm: An experimental album of Irish traditional music and computer-generated tunes,” KTH Royal Institute of Technology, Tech. Rep., 2018.
- [5] B. L. Sturm, “What do these 5,599,881 parameters mean? An analysis of a specific LSTM music transcription model, starting with the 70,281 parameters of its softmax layer,” in *Proc. Music Metacreation workshop of the Int. Conf. Computational Creativity*, 2018.
- [6] —, “How stuff works: LSTM model of folk music transcriptions,” in *Proc. Joint Workshop on Machine Learning for Music, ICML*, 2018.
- [7] K. Cho, B. van Merriënboer, Ç. Gülçehre, F. H. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *Proc. Conf. Empirical Methods in Natural Language Processing*, 2014.
- [8] N. Srivastava, G. Hinton, A. Krizhevsky, A. Krizhevsky, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from over-fitting,” *J. Machine Learning Research*, vol. 15, pp. 1929–1958, June 2014.
- [9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Neurocomputing: Foundations of research,” J. A. Anderson and E. Rosenfeld, Eds. Cambridge, MA, USA: MIT Press, 1988, ch. Learning Representations by Back-propagating Errors, pp. 696–699.
- [10] A. Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, 2012.
- [11] K. S. Tai, R. Socher, and C. D. Manning, “Improved semantic representations from tree-structured long short-term memory networks,” in *Int. Joint Conf. Natural Language Processing*, 2015.



# Modeling and Learning Rhythm Structure

Francesco Foscarin

CNAM, Paris

francesco.foscarin@cnam.fr

Florent Jacquemard

INRIA, Paris

florent.jacquemard@inria.fr

Philippe Rigaux

CNAM, Paris

philippe.rigaux@cnam.fr

## ABSTRACT

We present a model to express preferences on rhythmic structure, based on probabilistic context-free grammars, and a procedure that learns the grammars probabilities from a dataset of scores or quantized MIDI files. The model formally defines rules related to rhythmic subdivisions and durations that are in general given in an informal language. Rules preference is then specified with probability values. One targeted application is the aggregation of rules probabilities to qualify an entire rhythm, for tasks like automatic music generation and music transcription. The paper also reports an application of this approach on two datasets.

## 1. INTRODUCTION

In the context of music notation, rhythm is commonly modeled as a recursive subdivision of a temporal space organized in measures, beats and sub-beats. This naturally gives rise to a representation based on hierarchical structures (*aka* Rhythm Trees [1]). Moreover, this subdivision involves, at each level, choices based on the context (in particular the current metre) and on a long-established tradition of best practices. They can be expressed as rules such as the notation convention “*beam the notes in order to highlight the beat position*” [2]. Those rules (there are countless) express preferences on rhythm with different purposes (*e.g.* reduce complexity, improve readability, *etc.*), but remains at an informal level and their application is crafted in both the core of engraving software, and the expertise of their human users.

In the present paper we propose a formal framework to express these rules in a computational context that enables an automatic determination of rhythm structures. Our model is based on Probabilistic Context Free Grammars (PCFG), where production rules and attached weight values specify rhythmic subdivisions in a way that is both formal and close to the musical intuition. Parse trees, representing the grammars’ computations, also represent Rhythm Trees (hence score structure).

A PCFG acts as a model replacing informal notation rules. One could manually define such a model, but one can also *learn* this model, as soon as we dispose of a large enough, high quality training set (of parse trees).

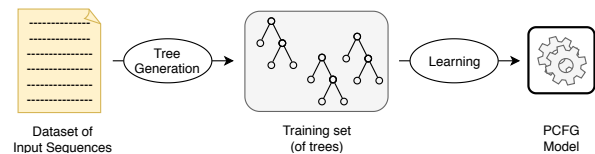


Figure 1. PCFG training from sequences of musical events.

An immediate thought is to base the learning step on the many corpora of existing (quantized) MIDI files or even digital scores. This gives rise however to an important issue: these datasets provide *sequences* of quantized events, but there is no direct mean to obtain the *hierarchical structures* (rhythm trees) that are necessary for learning a grammar. In the case of MIDI input, the rhythmic representation is simply missing (MIDI input). XML-based format such as MusicXML or MEI seem more suitable. However their hierarchical structure is not directly used for encoding rhythmic trees which makes their extraction unreliable. Moreover, there exists many ways to encode with these formats a same input, and this gives rise to ambiguities when it comes to identify a normalized rhythm representation.

In order to overcome this limitation, we propose to produce automatically training sets from collection of quantized input. Since there exists many possible rhythm trees that can be built from a single dataset entry, we need a decision guidelines to determine a unique candidate tree. Our decision method is based on the assumption that the *best rhythmic representation is the one that maximize the notation readability*. This assumption is supported by the analysis of music notation conventions, and corresponds to the intuition that the purpose of a notation language is to obtain a concise, accurate and readable representation of the noted content. The main goal of the present paper is to develop an algorithm for training set production based on this assumption, and to validate it on a set of representative datasets.

The production algorithm relies on the definition of a tree minimization criteria, and explores the space of solutions trees that correctly represent a sequence input in order to find the minimal one with respect to this criteria. Given a dataset of sequences, we then apply the algorithm to produce the corresponding training set of minimal rhythm trees, and then carry out maximum likelihood estimation in order to obtain the PCFG (Figure 1).

Finally, we validate our method by running our training algorithm on a set of representative corpus, checking that the obtained PCFG is consistent with best music notation practices. Our results confirm that the best tree decision

Copyright: © 2019 Francesco Foscarin et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

method based on tree minimization is a reliable computational method to produce PCFG that would, otherwise, have to be manually defined.

We believe that our methodology is important for several reasons. First, PCFGs are quite useful for music transcription. In a companion paper [3] we describe an algorithm that takes as input a non-quantized sequence of musical events (e.g., a human performance of some music work, with micro-rhythmic deviations) and relies on a PCFGs to produce a music score. Note, that a PCFG is, in this regard, more powerful than the tree-minimization method mentioned above that only operates on quantized inputs. Another common application for PCFGs is music generation [4]. More generally, disposing of a language model is useful to measure in which extent this model is a reliable representation of the actual language used in a corpus. In concrete terms, it can be used for instance to evaluate the quality of an existing notation, or to detect outliers in a corpus (e.g., scores of MIDI files that present an unusual rhythm). In general, we consider that this constitutes a quite useful analytic tool to make sense of sequential inputs that can be structured, quantified, explored and compared.

Lastly, the necessity of having a good grammar is fundamental to all the applications cited above. The technique presented in this paper allows to automatically build a grammar from a large dataset of scores or MIDI files, avoiding the manual building, a process that is very time consuming and is error prone.

To summarize, the paper: (1) uses PCFGs as a formalization of rhythm notation rules (Section 3), (2) learns PCFGs from datasets of music events sequences, producing training sets thanks to a complexity minimization criteria, (Section 4) and (3) validates that the resulting PCFGs trained on a dataset indeed accurately capture the best practices established in music notation (Section 5). We begin by Section 2 that briefly reviews some of the current works that use trees and grammar to work with rhythm and conclude with Section 6.

## 2. STATE OF THE ART

Many works in the literature rely on linear models (e.g., n-grams) that apply to the sequential flow of music events [5].

Another category, more suited to represent the hierarchical structure of rhythm notation, are models based on trees and grammars. Starting from the Generative Theory of Tonal Music by Lerdahl & Jackendoff [6], those models have been successfully explored for rhythmic notation processing and evaluation [1,7,8], meter detection [9], melodic search [10] and music analysis [11–14].

In [10], probabilistic tree automata (PTA) learning techniques are used for symbolic melody recognition. More precisely, given one melody  $M$  represented as a melodic tree (a structure similar to our parse trees), a PTA  $\mathcal{A}_M$  is computed, so that, when given another melody  $M$ ,  $\mathcal{A}_M$  will return the probability that  $M$  is a cover (or a variation, or a plagia) of  $M$ . Therefore, although the objectives of [10] differ from ours, a dataset of melodic trees was needed in this work. However, this dataset is small, and

can be constructed manually since, by definition, only 1 melody is needed in order to train 1 automaton.

In [15,16], a notation of rhythm languages defined by formal context-free grammars is proposed in order to fix the kinds of rhythmic notation to consider using declarative rules. In [3], we propose techniques based on weighted context-free grammars for automatic rhythm transcription, but the grammar is assumed given and no details are given about the procedure to construct it. In this paper we start from the same settings but we focus on the grammar creation, using results for context-free-grammars presented in [17] to obtain a model that can be trained on a dataset.

## 3. MODEL SPECIFICATION

Probabilistic Context-Free Grammars (PCFGs) extend CF grammars with rule probabilities. Computations of PCFGs are conveniently represented as hierarchical structures called *parse trees*. As observed in several papers, such tree structure are natural representations of common Western notation for rhythms, as they reflect structural nested decomposition of measures into beats.

### 3.1 Context Free Grammars for Rhythm

A PCFG is a tuple  $\mathcal{G} = \langle Q, q_{\text{init}}, R \rangle$  where (i)  $Q$  is a finite set of non-terminal symbols (*nt*), denoted  $q_0, q_1, \dots$ , (ii)  $q_{\text{init}} \in Q$  is a starting non-terminal, and (iii)  $R$  is a finite set of weighted production rules of one of the two following types, where  $w$  is a weight value in  $[0, 1]$ :

(*k-div*)  $q_0 \xrightarrow{w} q_1 \dots q_k$  with  $q_0, \dots, q_k \in Q$  and  $k > 1$ ,

(*leaf*)  $q_0 \xrightarrow{w} n$  with  $q_0 \in Q$  and  $n \in \mathbb{N}$ ,

such that for all  $q_0 \in Q$ ,  $\sum_{q_0 \xrightarrow{w} \alpha \in R} w = 1$  (where  $\alpha$  stands

for  $q_1, \dots, q_k \in Q$  or  $n \in \mathbb{N}$ ). The *nt*  $q_0$  is called the *head* of both above rules.

A production rule (*k-div*) describes the division of a time interval into parts of same length, e.g. the division of a quarter note into 2 eighth notes (for  $k = 2$ ) or into a triplet (for  $k = 3$ ). The recursive application of (*k-div*) rules represents nested divisions. A (*leaf*) rule expresses that the time interval  $I$  reached in *nt*  $q_0$  contains  $n$  events, all aligned at the left bound of  $I$ . When  $n > 1$ , it means that we have  $n - 1$  grace notes, of theoretical duration 0, followed by one note spanning over  $I$ . When  $n = 0$ ,  $I$  is called a *continuation*, and its function is similar to that of a tie or a dot in music notation. Continuations are a fundamental concept in our model, since they practically allow us to split a note in multiple parts that span multiple terminal symbols.

The weight of nested divisions and event alignments is the product of the weights of all the rules involved.

**Example 1.** Let us consider the PCFG in Table 1. Applying the rule  $\rho_1$  to  $[0, 1[$  results in two sub-intervals  $[0, \frac{1}{2}[$  and  $[\frac{1}{2}, 1[$ , and both of them can be processed with any rule with head  $q_{\frac{1}{2}}$ . Assume that we apply  $\rho_{11}$  to the first sub-interval and  $\rho_3$  to the second one, which is then divided

into  $[\frac{1}{2}, \frac{3}{4}[$  and  $[\frac{3}{4}, 1[$ . Then, we apply respectively  $\rho_{16}$  and  $\rho_{17}$  to the latter two sub-sub-intervals of length  $\frac{1}{4}$ . The above rule applications result in the division of the initial interval  $[0, 1[$  into a partition made of  $[0, \frac{1}{2}[$ ,  $[\frac{1}{2}, \frac{3}{4}[$  and  $[\frac{3}{4}, 1[$ . The first part contains a single event, at time 0, the second is a continuation (of the first event), and the last part contains a single event, at time  $\frac{3}{4}$ . Hence the above computation describes the rhythm represented in Figure 2.b.

Following our focus on rhythmic notation, the rules of type (leaf) only care about numbers of musical events, and contain no information about the events themselves, like pitch values, or the nature of events (note or chord). For a representation of melodies, one could replace the natural numbers in (leaf) rules by terminal symbols in some alphabet appropriate to the representation of musical events.

### 3.2 Parse Trees

We formalize the computations of a PCFG  $\mathcal{G} = \langle Q, q_{\text{init}}, R \rangle$  with the notion of *parse tree*, which is a tree  $t$  labeled with rules of  $R$ , such that: every inner node of  $t$  is labeled by a  $(k\text{-div})$  rule, every leaf of  $t$  is labeled by (leaf) rules, and if an inner node  $\eta$  is labeled by  $\rho = q_0 \xrightarrow{w} q_1 \dots q_k$ , then it has exactly  $k$  subtrees  $t_1, \dots, t_k$  whose respective roots have heads  $q_1, \dots, q_k$ . The subtree of  $t$  with root  $\eta$  is then denoted by  $\rho(t_1, \dots, t_k)$ . The weight  $\text{weight}(t)$  of a parse tree  $t$  is the product of the weights of all the transitions labeling its nodes. It is defined recursively by  $\text{weight}(\rho(t_1, \dots, t_k)) = w \times \prod_{i=1}^k \text{weight}(t_i)$  when  $\rho$  is  $q_0 \xrightarrow{w} q_1 \dots q_k$  and  $\text{weight}(\rho_0) = w$  for  $\rho_0 = q_0 \xrightarrow{w} n$ .

**Example 2.** The parse tree corresponding to the computation described in Example 1 is depicted in Figure 2.a.

### 3.3 Timelines and Parse Tree Serialization

We consider in the following time-points expressed in fraction of 1 measure (a rational value). A *timeline*  $\ell = \langle I, \sigma \rangle$  is the representation of a sequence of events made of a left-open time interval  $I = [p, p'[$  called *carrier* of  $\ell$  and a sequence  $\sigma$  of time-points inside  $I$ . We assume that  $\sigma$  is increasing but not strictly increasing (*i.e.* it may contain repetitions). Also, the first event in  $\sigma$  may be distinct from the left bound  $p$  of  $I$ . In this case (and also when  $\sigma$  is empty), it means that  $\ell$  starts with a continuation. The concatenation of two timelines  $\ell_1 = \langle [p_1, p'_1[, \sigma_1 \rangle$  and  $\ell_2 = \langle [p_2, p'_2[, \sigma_2 \rangle$  such that  $p_2 = p'_1$  is a timeline  $\ell = \langle [p_1, p'_2[, \sigma \rangle$  where  $\sigma$  is the concatenation of  $\sigma_1$  with  $\sigma_2$ .

We associate to every parse tree  $t$  of a PCFG  $\mathcal{G}$  and time interval  $I$  a timeline denoted  $\|t\|_I$  and defined by:

$$\|\rho_0\|_{[p, p'[} = \langle [p, p'[, \underbrace{(p, \dots, p)}_n \rangle \text{ for } \rho_0 = q_0 \xrightarrow{w} n, \text{ and}$$

$\|\rho(t_1, \dots, t_k)\|_I$  is the concatenation of the timelines  $\|t_1\|_{I_1}, \dots, \|t_k\|_{I_k}$ , for  $\rho = q_0 \xrightarrow{w} q_1 \dots q_k$ , and where  $I_1, \dots, I_k$  is a partition of  $I$  into  $k$  sub-intervals of equal duration.

**Example 3.** The parse tree  $t = \rho_1(\rho_{11}, \rho_3(\rho_{16}, \rho_{17}))$  of Figure 2.a is associated, for the time interval  $[0, 1[$ , the timeline represented in Figure 2.b, computed as follows

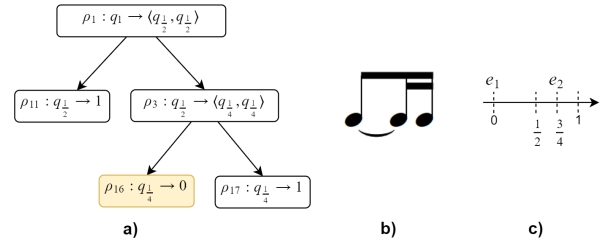


Figure 2. A parse tree (a), the respective music notation given a metric of  $\frac{1}{4}$  (b) and timeline given an interval  $[0, 1[$  (c). The leaf representing a *cont* is highlighted in yellow. Note that due to the *cont*, the timeline has 2 events, even if the parse tree has 3 leaves.

(the operator  $+$  denotes timeline concatenation):

$$\begin{aligned} \|t\|_{[0, 1[} &= \|\rho_{11}\|_{[0, \frac{1}{2}[} + \|\rho_3(\rho_{16}, \rho_{17})\|_{[\frac{1}{2}, 1[} \\ &= \|\rho_{11}\|_{[0, \frac{1}{2}[} + \|\rho_{16}\|_{[\frac{1}{2}, \frac{3}{4}[} + \|\rho_{17}\|_{[\frac{3}{4}, 1[} \\ &= \langle [0, \frac{1}{2}[, (0) \rangle + \langle [\frac{1}{2}, \frac{3}{4}[, () \rangle + \langle [\frac{3}{4}, 1[, (\frac{3}{4}) \rangle \\ &= \langle [0, 1[, (0, \frac{3}{4}) \rangle. \end{aligned}$$

We say that a parse tree  $t$  yields a timeline  $\ell = \langle I, \sigma \rangle$  iff  $\|t\|_I = \ell$ . Therefore every parse tree  $t$  of a PCFG  $\mathcal{G}$  yields an organization  $\|t_k\|_I$  of events in time and also a grouping structure for these events. In other terms,  $t$  is a consistent representation of music events with respect to the notation defined by  $\mathcal{G}$ , and given a time signature, a music score can be constructed from it (Figure 2). We call this process *score production*. Differently from the serialization process, the continuations remain in the final result of the score production.

A parse tree can be used to represent an entire score or part of it. In this paper we represent each measure of a score with a different parse tree, *i.e.* the timeline produced by the serialization of a parse tree will represent a single measure. To summarize, we use parse trees as a model for both rhythmic structure and rhythmic notation.

## 4. MODEL TRAINING

We consider the problem of computing weight values in the rules of a PCFG from a dataset made of timelines. Approaches based on maximum likelihood estimator [17] permit to obtain such weights from a training set made of parse trees. Therefore, in order to apply such approaches (in Section 4.2), we need to convert datasets of timelines into training sets containing parse trees (Section 4.1).

### 4.1 Training Set Construction from a Score Corpus

As mentioned in the introduction we produce a training set of parse trees from a dataset of monophonic sequences extracted from a corpus of scores. This applies to a wide range of input scores, from (quantized) MIDI files to XML scores. In the latter case, one could potentially benefit from the grouping elements (beaming and tuplets) in the music notation, but this information calls for high-quality corpora where the notation complies to the best practices. Our approach holds independently from such assumptions.

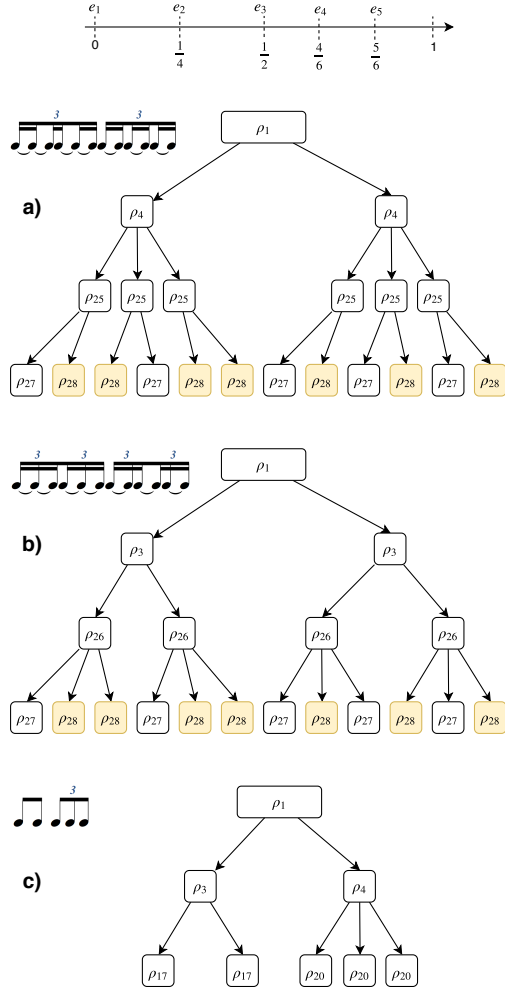


Figure 3. Three different trees for the same timeline. The terminal rules corresponding to a continuation are highlighted in yellow.

From each score (or MIDI file) in the corpus, each part in the score, each voice in the part, and each measure in the voice, we extract a timeline (of carrier  $[0, 1]$ ) from the list of event durations. We use this dataset  $\mathcal{D}$  of extracted timelines as input to build parse trees. The resulting set of parse trees is the training set  $\mathcal{T}$  used for learning a grammar.

Let us assume given an *acyclic* grammar  $\mathcal{G} = \langle Q, q_{\text{init}}, R \rangle$  whose weight are initially unknown (we call such  $\mathcal{G}$  *unweighted*).

We produce for each timeline  $\ell \in \mathcal{D}$  one parse tree  $t$  of  $\mathcal{G}$  such that  $\|t\|_{[0,1]} = \ell$ , called the *representative* of  $\ell$  in the training set. Since there exists several possible parse trees, we need a criteria to choose a unique representative.

We choose the tree  $t$  with a minimum number of leaves. This choice makes sense both from a computational and from a musical point of view. In fact, we prefer that the scores produced by  $\mathcal{G}$  not to be crowded with useless notes and ties. Later in Section 5.3 we will show that the results obtained with the above criteria are coherent with some common recommendations for rhythm notation. It means that the trained PCFG will be suited to represent

both rhythm and rhythm notation *wrt* such recommendations.

The following function *rep* returns for a *nt*  $q \in Q$  and a timeline  $\ell = \langle I, \sigma \rangle$ , a parse tree  $t$  of  $\mathcal{G}$ , with root headed by  $q$ , yielding  $\ell$ , and with a minimal number of leaves. In the definition of *rep*, the min of a set  $T$  of trees is the tree with a minimum number of leaves. This min is undefined when  $T$  is empty or contains at least two trees with a minimum number of leaves.

If  $\sigma$  is empty or all points of  $\sigma$  coincide with the left bound of  $I$ , then

$$\text{rep}(q_0, \ell) = \rho_0 \quad (1)$$

where  $\rho_0 = q_0 \rightarrow |\sigma|$ , if  $\rho_0 \in R$ , or else *rep*( $q_0, \ell$ ) is undefined. For the other cases of  $\sigma$ ,

$$\text{rep}(q, \ell) = \min_{\rho = q \rightarrow (q_1, \dots, q_k)} \left( \rho(\text{rep}(q_1, \ell_1), \dots, \text{rep}(q_k, \ell_k)) \right) \quad (2)$$

where  $\ell_1, \dots, \ell_k$  is the partition of  $\ell$  into  $k$  timelines of equal duration.

**Example 4.** Let us present some steps of the computation of *rep* for the grammar in Table 1 (forgetting the weight values), and the timeline  $\ell = \langle [0, 1[, (0, \frac{3}{4}) \rangle$  of Figure 2.

$$\text{rep}(q_1, \ell) = \min \begin{cases} \rho_1(\text{rep}(q_{\frac{1}{2}}, \ell_{2,1}), \text{rep}(q_{\frac{1}{2}}, \ell_{2,2})), \\ \rho_2(\text{rep}(q_{\frac{1}{3}}, \ell_{3,1}), \text{rep}(q_{\frac{1}{3}}, \ell_{3,2}), \text{rep}(q_{\frac{1}{3}}, \ell_{3,3})) \end{cases}$$

where

$$\begin{aligned} \ell_{2,1} &= \langle [0, \frac{1}{2}[, (0) \rangle, \ell_{2,2} = \langle [\frac{1}{2}, 1[, (\frac{3}{4}) \rangle, \\ \ell_{3,1} &= \langle [0, \frac{1}{3}[, (0) \rangle, \ell_{3,2} = \langle [\frac{1}{3}, \frac{2}{3}[, ( ) \rangle, \\ \ell_{3,3} &= \langle [\frac{2}{3}, 1[, (\frac{3}{4}) \rangle. \end{aligned}$$

Following (1),  $\text{rep}(q_{\frac{1}{2}}, \ell_{2,1}) = \rho_{11}$ ,  $\text{rep}(q_{\frac{1}{3}}, \ell_{3,1}) = \rho_{14}$ , and  $\text{rep}(q_{\frac{1}{3}}, \ell_{3,2}) = \rho_{13}$ . For  $\text{rep}(q_{\frac{1}{2}}, \ell_{2,2})$  and  $\text{rep}(q_{\frac{1}{3}}, \ell_{3,3})$ , more computation steps are needed.

The function *rep* can be implemented efficiently with Dynamic Programming through a tabulation procedure similar to the CYK parsing algorithm [18].

The representative of a 1-measure timeline  $\ell$  in the dataset  $\mathcal{D}$  is  $\text{rep}(q_{\text{init}}, \ell)$ . It may be undefined, either because there is no parse tree  $t$  of  $\mathcal{G}$  yielding  $\ell$ , or, on the contrary, because there are more than one such parse trees of  $\mathcal{G}$  with a minimum number of leaves. In the first case,  $\mathcal{G}$  is too small to represent  $\mathcal{D}$  and should be completed. The second case is discussed in Section 4.2.

This simple definition of *rep* is correct because the function which associate to a tree its number of leaves is monotonic, *i.e.* if  $t_i$  has more leaves than  $t'_i$ , then  $\rho(t_1, \dots, t_i, \dots, t_k)$  has more leaves than  $\rho(t_1, \dots, t'_i, \dots, t_k)$ . It follows that we can build a best representative  $t$  for a timeline  $\ell$  (*wrt* this criteria) from best representative for sub-timelines of  $\ell$  (which are sub-trees of  $t$ ).



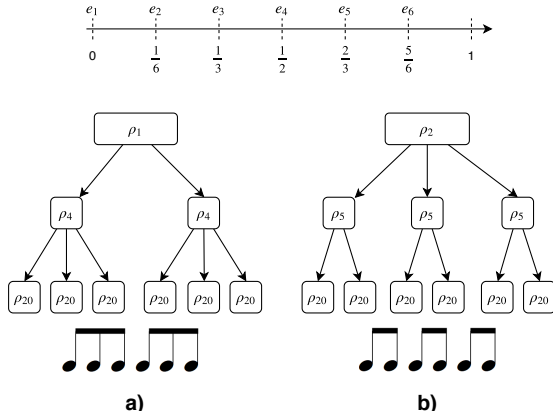


Figure 4. Two trees yielding the same timeline and corresp. notation in time signature  $\frac{3}{4}$  or  $\frac{6}{8}$ . Both have a minimal number of leaves 6 and we cannot choose a representative between them.

#### 4.2 Computation of Grammar's Weights

Let  $\mathcal{T}$  be our training set of parse trees. We then compute the weight values for the rules  $\mathcal{G}$  with a *Maximum Likelihood Estimator* [17]. For a rule  $\rho = q_0 \rightarrow \alpha$ , where  $\alpha$  is either  $q_1, \dots, q_k$  or  $n \in \mathbb{N}$ , let  $C_{\mathcal{T}}(\rho)$  be the number of occurrences of  $\rho$  in the trees of  $\mathcal{T}$ . The weight value for the rule  $\rho$  is then defined as  $\frac{C_{\mathcal{T}}(\rho)}{\sum_{q_0 \rightarrow \beta \in R} C_{\mathcal{T}}(q_0 \rightarrow \beta)}$ .

One can check that the grammar  $\mathcal{G}'$  defined from  $\mathcal{G}$  with these weight values is a PCFG, according to the definition in Section 3.1 (see [17]).

Given a timeline  $\ell \in \mathcal{D}$ , it may happen that its representative is undefined because there is more than one parse tree yielding  $\ell$  with minimum number of leaves, see the example in Figure 4. In this case, it is not possible to choose a unique representative, and we will initially discard such  $\ell$ . However, those timelines may contains useful information, and we propose the following two step procedure:

1. compute the training set  $\mathcal{T}$  like in in Section 4.1, using only the timelines of  $\mathcal{D}$  with a defined (unique) representative, and define weight values for  $\mathcal{G}$  from  $\mathcal{T}$  as above, resulting in a PCFG  $\mathcal{G}'$ .
2. for every timeline  $\ell \in \mathcal{D}$  discarded at step 1, compute a representative  $t$  which is a parse tree of  $\mathcal{G}'$  build with a modification of the function of Section 4.1 where the min wrt the number of leaves of  $t$  is replaced by the max of  $weight(t)$ . Compute new weight values with these representatives, resulting in a new PCFG  $\mathcal{G}''$ .

### 5. IMPLEMENTATION

In this section we first propose two different families of unweighted acyclic grammars acting as input to the construction of Section 4, then we present the result of the grammar learning algorithm from some score corpora along with some consideration from a musical perspective.

$\rho_1 : q_1 \xrightarrow{0.6} \langle q_{\frac{1}{2}}, q_{\frac{1}{2}} \rangle$	$\rho_2 : q_1 \xrightarrow{0.2} \langle q_{\frac{1}{3}}, q_{\frac{1}{3}}, q_{\frac{1}{3}} \rangle$
$\rho_3 : q_{\frac{1}{2}} \xrightarrow{0.1} \langle q_{\frac{1}{4}}, q_{\frac{1}{4}} \rangle$	$\rho_4 : q_{\frac{1}{2}} \xrightarrow{0.7} \langle q_{\frac{1}{6}}, q_{\frac{1}{6}}, q_{\frac{1}{6}} \rangle$
$\rho_5 : q_{\frac{1}{3}} \xrightarrow{0.6} \langle q_{\frac{1}{6}}, q_{\frac{1}{6}} \rangle$	$\rho_6 : q_{\frac{1}{3}} \xrightarrow{0.3} \langle q_{\frac{1}{9}}, q_{\frac{1}{9}}, q_{\frac{1}{9}} \rangle$
$\rho_7 : q_1 \xrightarrow{0.05} 0$	$\rho_8 : q_1 \xrightarrow{0.1} 1$
$\rho_9 : q_1 \xrightarrow{0.05} 2$	$\rho_{10} : q_{\frac{1}{2}} \xrightarrow{0} 0$
$\rho_{11} : q_{\frac{1}{2}} \xrightarrow{0.1} 1$	$\rho_{12} : q_{\frac{1}{2}} \xrightarrow{0.1} 2$
$\rho_{13} : q_{\frac{1}{3}} \xrightarrow{0.05} 0$	$\rho_{14} : q_{\frac{1}{3}} \xrightarrow{0.05} 1$
$\rho_{15} : q_{\frac{1}{3}} \xrightarrow{0.0} 2$	$\rho_{16} : q_{\frac{1}{4}} \xrightarrow{0.1} 0$
$\rho_{17} : q_{\frac{1}{4}} \xrightarrow{0.8} 1$	$\rho_{18} : q_{\frac{1}{4}} \xrightarrow{0.1} 2$
$\rho_{19} : q_{\frac{1}{6}} \xrightarrow{0.3} 0$	$\rho_{20} : q_{\frac{1}{6}} \xrightarrow{0.7} 1$
$\rho_{21} : q_{\frac{1}{6}} \xrightarrow{0} 2$	$\rho_{22} : q_{\frac{1}{9}} \xrightarrow{0.5} 0$
$\rho_{23} : q_{\frac{1}{9}} \xrightarrow{0.3} 1$	$\rho_{24} : q_{\frac{1}{9}} \xrightarrow{0.2} 2$

Table 1. An example of grammar with  $K_{max} = 3$ ,  $D_{max} = 2$  and  $gn_{max} = 2$ . Rules 1 to 9 are  $k$ -div rules and from 10 to 24 they are leaf rules.

#### 5.1 Use-Cases of Unweighted Grammar

In theory, the learned grammar (the one given to the learning step, composed of rules without weights) must be as complete as possible. However, for practical reasons and in particular the size of a grammar that would represent *all* possible rules, we need to adopt some restrictions. In our implementation, we chose to:

1. allow  $k$ -div rules only with  $k$  prime number, up to a prime number  $K_{max}$ . Other  $k$ -split can be obtained by sequential splits by the prime-number factors of  $k$ , e.g. to represent a 6-tuplet, we split by 2 and 3.
2. allow sequential rule application up to a maximum depth  $D_{max}$ , e.g. with  $D_{max} = 2$  we can split an interval in  $k$  sub-interval, recursively split each one of them and then stop the recursion.
3. allow only  $gn_{max}$  events in a interval, i.e.  $gn_{max} - 1$  grace-notes and one general-note.

The following are examples of practical PCFGs that respect these choices.

**Example 5.** A first possibility (see Table 1) is to define on rules that do not distinguish intervals of equal size in a measure, whatever their position. Thus,  $[0, \frac{1}{2}[$  and  $[\frac{1}{2}, 1[$  are represented by the same non terminal  $q_{\frac{1}{2}}$ . Each non terminal symbol represents a time interval of a specific duration, and the terminals productions specify how many events are contained in that interval (aligned to the left boundary). Given the grammar of Table 1 an informal notational rule for a  $\frac{6}{8}$  metric like: “prefer to divide in 2 parts at measure level and subsequently in 3 parts” will translate in our framework in  $weight(\rho_1) \geq weight(\rho_2)$  and  $weight(\rho_4) \geq weight(\rho_3)$ .

The grammar above is reduced in size, but does not allow for fine-grained distinction of rules based on the position of an interval in a measure, e.g., starting on a strong beat or not. Another possibility is given below.

**Example 6.** Another possibility is to use a larger set of non-terminal symbols that can distinguish intervals both on the level of recursion and on the horizontal position. For instance, the first half of a measure can be treated differently from the second half, or grace notes (in (leaf) rules) can be allowed for the first note of a tuplet and forbidden for the others. It allows to formally represent rules such as: “Prefer to have longer notes on stronger beats” (given that the time intervals that correspond to “stronger beats” are known), by assigning a higher probability to a leaf rule in those intervals and a higher probability of a  $k$ -div rule to the other intervals at the same level.

## 5.2 Score Corpora and Datasets

We trained the two grammars presented in the above section with the 1-measure timelines extracted from two corpora of scores: Music21 corpus<sup>1</sup> and Enhanced Wikifonia Leadsheet Dataset (EWLD) dataset [19]. We could compute grammars for the whole dataset but there exist subsets of scores sharing some common properties that are likely to yield more consistent grammars if they are processed independently. One such property is for instance the time signature. Other possible groups could be inferred by style, tempo marking and author, depending of the level of precision that is required. We chose to divide our datasets in four subsets defined by the following time signatures:  $\frac{4}{4}$ ,  $\frac{3}{4}$ ,  $\frac{6}{8}$ ,  $\frac{12}{8}$ . The number of scores for each group is reported in Table 2.

Within each score, we performed a simple operation of data cleaning, deleting the measure whose events durations did not sum to the correct duration given by the signature (*i.e.* pickup measures), final measures or incorrectly notated measures.

## 5.3 Trained PCFG from a Notational Point of View

In this section we analyze the result of the training step from a musical point of view in order to show that the criteria that we used to build unique parse trees from durations (Section 4.1) is coherent with music notation conventions.

From music general conventions [2] we know that different time signatures have different grouping preferences in order to expose the beat (*e.g.* points were stronger accent are placed). It is interesting to notice that there is an high correlation between the probabilities learned for our grammar (the grammar of the Example 5 is sufficient for this analysis) and the divisions suggested by music notation (Table 3).

For example, music conventions state that a  $\frac{3}{4}$  measure should be divided first in 3 parts (3 quarter notes); We can translate this rules in a grammar-form assigning a higher probability to the 3-div of a measure (with respect to other  $k$ -div at measure level). Looking at the trees that we produced from the  $\frac{3}{4}$  measures (with the algorithm in Section 4.1), we notice that this notation convention is respected, since our trees have a 3-div at measure level in 82% of the cases.

<sup>1</sup> <http://web.mit.edu/music21/doc/about/referenceCorpus.html>

Our criteria to find the smallest tree (minimization of *yield*), allows us to build trees that are coherent with notation convention; therefore our model make sense to express rules about both rhythmic structure and rhythmic notation. From another point of view we can say that our criteria of minimizing the leaves generates results similar to that of an expert engraver who aims at making the notation as readable as possible.

## 6. CONCLUSIONS AND FUTURE WORKS

In this paper we presented a model of rhythm structure based on context-free-grammars and a way to learn it from a dataset of scores, addressing the problem of the generation of a training set of trees. We show that our model makes sense both from rhythm structure point of view than from a rhythmic notation point of view, comparing the frequency of the divisions in our model, with suggested divisions in music notation.

The complete implementation<sup>2</sup> of the grammar generation and learning is made in python and C++. The system is also partially implemented in the online digital score library NEUMA, in order to learn grammars from the corpus of scores online.

With big grammars (particularly with the complete grammar), we still have the problems of sparsity, *i.e.* we have lots of zeros in the final results. The best solution would be to have a bigger dataset, but alternatively the typical approach is to use a smoothing technique. However we would need to think carefully how to apply the smoothing in order to add probabilities to rare rules in a way that makes sense from a musical perspective.

The next objective is to test those grammar in a music transcription algorithm from a MIDI performance, in order to retrieve at the same time the quantized performance and the relative score.

## Acknowledgments

This work was partially funded by the ANR project MUNIR. The authors want to thank Masahiko Sakai for his comments and suggestions.

## 7. REFERENCES

- [1] C. Agón, K. Haddad, and G. Assayag, “Representation and rendering of rhythm structures,” in *Second International Conference on WEB Delivering of Music (WEDELMUSIC)*, 2002, pp. 109–116.
- [2] E. Gould, *Behind bars*. Faber Music, 2011.
- [3] F. Foscarin, F. Jacquemard, P. Rigaux, and M. Sakai, “A Parse-based Framework for Coupled Rhythm Quantization and Score Structuring,” Jan. 2019, working paper or preprint. [Online]. Available: <https://hal.inria.fr/hal-01988990>

<sup>2</sup> Available at <https://github.com/fosfrancesco/Modeling-Rhythm-Structure.git>.

	music21 corpus				EWLD			
	scores	measures	discarded	failed	scores	measures	discarded	failed
4/4	3261	105684	17506	139	3544	127436	3067	11554
3/4	1875	55817	9918	3228	670	30583	454	8619
6/8	2161	61048	10039	478	101	4342	44	586
12/8	9	417	129	7	129	770	10	30

Table 2. The number of scores and measures for each corpus and each time signature considered. The discarded measures are the measures whose sum does not correspond to the correct time signature; the failed measures are the one for which it was not possible to build a tree representative.

		music21	EWLD
4/4	1st div. by 2	99%	99%
	2nd div. by 2	98%	99%
	3rd div. by 2	93%	96%
3/4	1st div. by 3	82%	91%
	2nd div. by 2	99%	96%
6/8	1st div. by 2	70%	60%
	2nd div. by 3	90%	91%
12/8	1st div. by 2	74%	98%
	2nd div. by 2	60%	62%
	3rd div. by 3	71%	93%

Table 3. Frequency of divisions suggested in music notation in the sets of trees built from our datasets of scores. Note that in this table the frequency have been normalized on the possible divisions, not considering terminal productions.

- [4] D. Conklin, "Music generation from statistical models," in *AISB Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, 2003, pp. 30–35.
- [5] H. Takeda, N. Saito, T. Otsuki, M. Nakai, H. Shimodaira, and S. Sagayama, "Hidden markov model for automatic transcription of midi signals," in *IEEE Workshop on Multimedia Signal Processing*. IEEE, 2002, pp. 428–431.
- [6] F. Lerdahl and R. S. Jackendoff, *A generative theory of tonal music*. MIT press, 1985.
- [7] P. Nauert, "A theory of complexity to constrain the approximation of arbitrary sequences of timepoints," *Perspectives of New Music*, pp. 226–263, 1994.
- [8] F. Jacquemard, P. Donat-Bouillud, and J. Bresson, "A structural theory of rhythm notation based on tree representations and term rewriting," in *Mathematics and Computation in Music*. Springer, 2015, pp. 3–15.
- [9] A. McLeod and M. Steedman, "Meter detection and alignment of midi performance." International Conference on Music Information Retrieval (ISMIR), 2018.
- [10] J. F. Bernabeu, J. Calera-Rubio, J. M. Iñesta, and D. Rizo, "Melodic identification using probabilistic tree automata," *Journal of New Music Research*, vol. 40, no. 2, pp. 93–103, 2011.
- [11] M. Granroth-Wilding and M. Steedman, "Statistical parsing for harmonic analysis of jazz chord sequences," in *Proc. Intl. Computer Music Conference (ICMC)*, 2012.
- [12] M. Rohrmeier, "Towards a generative syntax of tonal harmony," *Journal of Mathematics and Music*, vol. 5, no. 1, pp. 35–53, 2011.
- [13] E. Nakamura, M. Hamanaka, K. Hirata, and K. Yoshii, "Tree-structured probabilistic model of monophonic written music based on the generative theory of tonal music," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 276–280.
- [14] A. Marsden, S. Tojo, and K. Hirata, "No longer'somewhat arbitrary': calculating salience in gttm-style reduction," in *Proceedings of the 5th International Conference on Digital Libraries for Musicology*. ACM, 2018, pp. 26–33.
- [15] F. Jacquemard, A. Ycart, and M. Sakai, "Generating equivalent rhythmic notations based on rhythm tree languages," in *International Conference of Technologies for Music Notation and Representation (TENOR)*, 2017.
- [16] A. Ycart, F. Jacquemard, J. Bresson, and S. Staworko, "A supervised approach for rhythm transcription based on tree series enumeration," in *International Computer Music Conference (ICMC)*, 2016.
- [17] Z. Chi and S. Geman, "Estimation of probabilistic context-free grammars," *Computational linguistics*, vol. 24, no. 2, pp. 299–305, 1998.
- [18] T. Kasami, "An efficient recognition and syntax-analysis algorithm for context-free languages," Air Force Cambridge Research Lab, Bedford, MA, Tech. Rep. Scientific report AFCRL-65-758, 1965.
- [19] F. Simonetta, "Enhanced wikifonia leadsheet dataset," Nov. 2018. [Online]. Available: <https://doi.org/10.5281/zenodo.1476555>

---

# Acknowledgments

The 16th Sound & Music Computing Conference (SMC 2019) was made possible thanks to the hard work of many people including the authors, the reviewers, all the members of the Conference Committee and other collaborators.

Special thanks go to the Conference Sponsors:

- Platinum Sponsors:
  - Universidad de Málaga, Andalucía Tech.
- Gold Sponsors:
  - FAST: Fusing Audio and Semantic Technologies For Intelligent Music Production and Consumption.
- Silver Sponsors:
  - Audio-Technica Corporation.
- Bronze Sponsors:
  - Applied Sciences, MDPI.
  - The Nordic Sound and Music Computing Network (Nordic SMC).
  - Fundación Unicaja.

The SMC 2019 Conference is possible only thanks to the excellent contribution of the SMC community. The biggest acknowledgment goes to you, the authors, researchers, musicians and participants of this conference.

## **General Chairs SMC 2019**

*Isabel Barbancho*  
*Lorenzo J. Tardón*

## **Program Chairs SMC 2019**

*Stefania Serafin*  
*Federico Avanzini*





---

# Reviewers

Andrade Esquef, Paulo Antonio  
Aramaki, Mitsuko  
Armitage, Jack  
Balke, Stefan  
Bank, Balázs  
Baratè, Adriano  
Barumerli, Roberto  
Barumerli, Roberto  
Bauer, Christine  
Benveniste, Samuel  
Bishop, Laura  
Bishop, Christopher  
Bonev, Martin  
Brattico, Elvira  
Bresin, Roberto  
Bryan-Kinns, Nick  
Burloiu, Grigore  
Cambouropoulos, Emilios  
Campbell, Tom  
Cancino Chacon, Carlos Eduardo  
Cardoso, F. Amílcar  
Carpentier, Thibaut  
Cavdir, Doga  
Chafe, Chris  
Chemla, Axel  
Chourdakis, Emmanouil Theofanis  
Collecchia, Regina  
Cossettini, Luca  
Dannenberg, Roger  
Das, Orchisama  
Davanzo, Nicola  
Davies, Matthew  
Delle Monache, Stefano  
Donin, Nicolas  
Drioli, Carlo  
Ducceschi, Michele  
Erkut, Cumhur  
Esqueda, Fabian  
Essl, Georg  
Fernandez, Jose Miguel  
Fontana, Federico  
Gan, Woon-Seng

Gasser, Martin  
Georgaki, Anastasia  
Gkiokas, Aggelos  
Goebel, Werner  
Grachten, Maarten  
Grani, Francesco  
Hanna, Pierre  
Hansen, Kjetil Falkenberg  
Holonowicz, Piotr  
Horner, Andrew  
Hug, Daniel  
Jouvelot, Pierre  
Katayose, Haruhiro  
Kelz, Rainer  
Kontogeorgakopoulos, Alexandros  
Kostek, Bozena  
Lähdeoja, Otso  
Lartillot, Olivier  
Leonard, James  
Lepri, Giacomo  
Lokki, Tapio  
Lukashevich, Hanna  
Lunn, Paul  
Lympouridis, Vangelis  
Maestre, Esteban  
Mandanici, Marcella  
Marolt, Matija  
Marozeau, Jeremy  
Matuszewski, Benjamin  
Mauro, Davide Andrea  
Mesaros, Annamaria  
Misdariis, Nicolas  
Mitchell, Thomas  
Morreale, Fabio  
Nam, Juhan  
Nilsson, Niels Christian  
Ntalampiras, Stavros  
Orlarey, Yann  
Paiva, Rui Pedro  
Papetti, Stefano  
Parker, Julian  
Pauwels, Johan

Perez-Carrillo, Alfonso  
Poirot , Samuel  
Polotti, Pietro  
Ponce De León, Pedro J.  
Presti, Giorgio  
Pretto, Niccolò  
Rocchetto, Davide  
Roze, David  
Schaffert, Nina  
Schwarz, Diemo  
Selfridge, Rod  
Sguerra , Lucas  
Simonetta, Federico  
Siwiak, Diana  
Smith, Julius  
Spagnol, Simone  
Stewart, Rebecca

Stockman, Tony  
Takala, Tapio  
Tamer, Burak  
Tillmann, Barbara  
Tiraboschi, Marco  
Trail, Shawn  
Turchet, Luca  
Tzanetakis, George  
Valimaki, Vesa  
Vicinanza, Domenico  
Villeneuve, Jerome  
Wanderley, Marcelo  
Werner, Kurt  
Willemsen, Silvin  
Ystad, Solvi  
Zangerle, Eva

---

# Author Index

- Çamcı, Anıl, 195
- Abeßer, Jakob, 551  
Adjorlu, Ali, 261  
Ahlbäck, Sven, 519  
Anagnostopoulou, Christina, 476  
Anderson, Nikolaj, 95, 151  
Andersson, Nikolaj S., 210  
Arro, Joann Gustav, 324  
Arthur, Claire, 297, 407  
Atienza, Ricardo, 119  
Avanzini, Federico, 372
- Baarlink, Johanna Friederike, 501  
Barahona, Adrian, 34  
Baratè, Adriano, 372  
Barbancho, Ana M., 93, 295, 490, 492  
Barbancho, Isabel, 93, 293, 295, 490, 492  
Basaran, Dogac, 537  
Bauman, Christian, 501  
Bednarz, Tomasz, 269  
Benetos, Emmanouil, 545  
Bergsland, Andreas, 39  
Bernardes, Gilberto, 348  
Betancourt Barriga, Nathaly Belen, 261  
Bevan, Ann, 281  
Bilbao, Stefan, 95, 151  
Bimbot, Frédéric, 423  
Bisig, Daniel, 26  
Björnsson, Sverrir Karl, 55  
Blackmore, Simon, 494  
Bozkurt, Baris, 72  
Brea, Johanni, 380  
Bresin, Roberto, 167, 255  
Bresson, Jean, 84  
Brown, Jonathon, 297  
Bruun-Pedersen, Jon Ram, 348  
Byrne, Derek Victor, 300
- Cano, Estefania, 299  
Cartledge, David, 387  
Cavdir, Doga, 105  
Chadna, Pritish, 447  
Chamorro, África, 490  
Chapman, Duncan, 310
- Chew, Elaine, 545  
Christensen, Pelle Juul, 234  
Clark, Beach, 407  
Cobos, Oscar, 295  
Colombo, Florian, 380  
Crayencour, Hélène-Camille, 537  
Cuesta, Helena, 447
- Dahl, Sofia, 159, 241  
Dalton, Brett, 7  
Damian, Daniela, 202  
Damskägg, Eero-Pekka, 332  
Dannenberg, Roger, 65  
Davis, Tom, 281  
Demirel, Emir, 72  
Dorigatti, Enrico, 86  
Dubois, Juliette, 530
- Eigenfeldt, Arne, 287  
Elowsson, Anders, 530  
Eni, Mihai, 310
- Fasciani, Stefano, 97  
Fierro, David, 61  
Fierro, Leonardo, 135  
Fober, Dominique, 112, 182  
Fontana, Federico, 210  
Foscarin, Francesco, 566  
Friberg, Anders, 530  
Frid, Emma, 167, 255  
Fujii, Junko, 289
- Gómez, Emilia, 447  
Gómez-Plazas, Irene, 93  
Georgaki, Anastasia, 476  
Gerry, Lynda Joy, 241, 495  
Gerstner, Wulfram, 380  
Gibson, Darrell, 302  
Gillot, Valentin, 423  
Girin, Laurent, 415  
Goto, Masataka, 505  
Grandjean, Didier, 174  
Greene, John, 297  
Grollmisch, Sascha, 299, 551



- Høeg, Emil Rosenlund, 348  
 Hallström, Eric, 558  
 Hamanaka, Masatoshi, 82  
 Hansen, Kjetil Falkenberg, 119  
 Henson, Jared, 297  
 Hirata, Keiji, 401  
 Hoby, Mads, 499  
 Holland, Davis, 310  
 Holzapfel, Andre, 274, 519  
 Hsu, Jennifer, 356, 454  
 Hu, Jamin, 143  
 Hueber, Thomas, 415  
 Huzaifah, Muhammad, 525  
  
 Järveläinen, Hanna, 511  
 Jacquemard, Florent, 566  
 Jaime, Víctor, 89  
 Jap, Lilian, 274  
 Jensenius, Alexander Refsum, 217  
 Jiang, Yucong, 387, 394  
 Jiang, Zheng, 65  
 Johnson, David, 7  
 Johnson, David, 202  
 Jurado-Navas, Antonio, 293  
 Juvela, Lauri, 332  
  
 Kalonaris, Stefano, 12  
 Kantan, Prithvi, 159  
 Kartofelev, Dmitri, 324  
 Keenan, Fiona, 127  
 Kelkar, Tejaswinee, 495  
 Kitahara, Tetsuro, 289  
 Klissouras, Odysseas, 20  
 Koepke, A. Sophia, 483  
 Kontogeorgakopoulos, Alexandros, 20  
 Kreković, Gordan, 316  
 Kuchelmeister, Volker, 269  
  
 Lan, Qichao, 217  
 Landsnes, Kristoffer, 250  
 Landy, Leigh, 310  
 Lartillot, Olivier, 174, 489  
 Latupeirissa, Adrian B., 255  
 Lem, Nolan, 470  
 Leonard, James, 187, 340  
 Letz, Stéphane, 112, 182  
 Lieck, Robert, 250  
 Limier, Samuel, 415  
 Lin, Kin Wah Edward, 505  
 Ljungdahl-Eriksson, Martin, 119  
 Ludovico, Luca A., 372  
  
 Müller, Meinard, 47  
 Maezawa, Akira, 364  
 Magalhaes, Eduardo, 348  
 Mandanici, Marcella, 372  
 Martin, Vincent P., 537  
 Mathiesen, Signe Lund, 300  
 McCoy, Eamon, 297  
  
 Mehrabyan, Liana, 250  
 Meihui, Tong, 223  
 Meng, Xiaojie, 545  
 Michon, Romain, 112, 182  
 Milde, Jan-Torsten, 501  
 Misgeld, Olof, 519  
 Molina Villota, Daniel Hernán, 293  
 Moreno, Daniel, 492  
 Moreno, Josué, 442  
 Moschos, Fotis, 61  
 Moss, Fabian C., 250  
 Mossmyr, Simon, 558  
 Moth-Poulsen, Mie, 269  
 Mounsbach, Martin, 431  
 Munilla, Jorge, 295  
  
 Nadar, Christon-Ragavan, 551  
 Nakano, Tomoyasu, 505  
 Namboodiri, Vinay P., 291  
 Neff, Patrick, 26  
 Neupert, Max, 80  
 Nilsson, Niels C., 210  
 Nordahl, Rolf, 210, 348  
 Norilo, Vesa, 442  
 Noto, Kaede, 401  
  
 Orlarey, Yann, 112, 182  
  
 Paisa, Razvan, 210  
 Panariello, Claudio, 167  
 Panda, Swaroop, 291  
 Passalenti, Andrea, 210  
 Pauletto, Sandra, 34, 127  
 Pauwels, Johan, 497  
 Pearce, Stephen, 310  
 Pecquet, Frank, 61  
 Pecquet, Justin, 61  
 Peinado, Alberto, 89  
 Peter, Silvan Davis, 88  
 Pierson, Daniel, 281  
 Pinder, James, 297  
 Polfreman, Richard, 302  
 Purkhús, Kristján Bjarki, 55  
  
 Rämö, Jussi, 135  
 Raphael, Christopher, 387, 394  
 Reynal, Sylvian, 537  
 Rigaux, Philippe, 566  
 Roche, Fanny, 415  
 Rohrmeier, Martin, 250  
 Roy, Shatarupa Thakurta, 291  
 Ryan, Fiona, 387  
  
 Sandler, Mark B., 497  
 Santini, Giovanni, 229  
 Saue, Sigurd, 39  
 Schacher, Jan, 26  
 Schreiber, Hendrik, 47

Serafin, Stefania, 95, 151, 210, 234, 241, 261, 269, 348, 431  
 Serra, Xavier, 72  
 Shafiei, Sepideh, 437  
 Sioros, George, 20  
 Sköld, Mattias, 167  
 Smyth, Tamara, 356, 454  
 Spagnol, Simone, 55  
 Stokke, Pekka, 39  
 Sturm, Bob L., 558

Tørresen, Jim, 217  
 Takegawa, Yoshinari, 401  
 Tardón, Lorenzo J., 93, 295, 490, 492  
 Tojo, Satoshi, 223  
 Tuovinen, Joonas, 143  
 Tzanetakis, George, 7, 202

Unnthórsson, Runar, 55

Välimäki, Vesa, 135, 143, 324, 332  
 Valle, Andrea, 462  
 Varela-Salinas, María José, 93  
 Vassilakis, Dimitrios, 476  
 Vegeborn, Victor Hansjongs, 558  
 Villena-Rodríguez, Alejandro, 93  
 Villeneuve, Jerome, 187, 340  
 Vinjar, Anders, 84  
 Vohra, Manohar, 97

Wang, Changhong, 545  
 Wang, Qian Janice, 300  
 Wedin, Jonas, 558  
 Wegener, Clemens, 80  
 Widmer, Gerhard, 88  
 Wiklund, Victor, 250  
 Wiles, Olivia, 483  
 Willemsen, Silvin, 95, 151  
 Wu, Fu-Hai Frank, 91  
 Wyse, Lonce, 525

Yasuhara, Akane, 289

Zisserman, Andrew, 483



---

## ORGANIZATION



UNIVERSIDAD  
DE MÁLAGA

| **uma.es**



---

## SPONSORS

### Platinum Sponsors



UNIVERSIDAD  
DE MÁLAGA



Vicerrectorado de Investigación y Transferencia

### Gold Sponsors



### Silver Sponsors



**audio-technica**

### Bronze Sponsors



**applied sciences**  
an Open Access Journal by MDPI



---

### Other collaborators



Proceedings of the 16th Sound & Music Computing Conference  
ISBN 978-84-09-08518-7 ISSN 2518-3672





28 - 31 May  
16th Sound and Music Computing Conference

### Conference Program

	Tuesday, May 28	Wednesday, May 29	Thursday, May 30	Friday, May 31
09:00		REGISTRATION		
09:30	WiSMC	WELCOME	ORAL S3	ORAL S6
10:00		KEYNOTE 1		
10:30			COFFEE P2 & D2	COFFEE P3 & D3
11:00		COFFEE P1 & D1	KEYNOTE 2	KEYNOTE 3
11:30				
12:00		ORAL S1	ORAL S4	ORAL S7
13:00		LUNCH POSTER P1 & DEMO D1	LUNCH POSTER P2 & DEMO D2	LUNCH POSTER P3 & DEMO D3
14:30		ORAL S2	ORAL S5	MUSIC M1
16:30				ORAL S8
17:00				MUSIC M2
18:30				
19:00	REGISTRATION	CONCERT		
19:30		MUSIC SMC		
20:00	WELCOME RECEPTION		DINNER	
				AWARDS & CLOSING



978-84-09-08518-7

ISSN 2518-3672